LA-UR-05-8228

Approved for public release; distribution is unlimited.

Title:

NON-LINEAR ERROR ANSATZ MODELS FOR SOLUTION VERIFICATION IN COMPUTATIONAL PHYSICS (U)

Internal Report of FY05 ASC Level-5 Milestone "VV-L5-JA4B-14"

Author(s): François M. Hemez, Los Alamos National Laboratory, X-7

Submitted to: The ASC Verification and Validation (V&V) Program Office
Los Alamos National Laboratory, Los Alamos, New Mexico 87545

Date: October 20, 2005

This page is left blank intentionally.

“Linearity breeds contempt.” — Peter Lax

This page is left blank intentionally.

Non-linear Error Ansatz Models for Solution Verification in Computational Physics

Report for FY05 ASC Level-5 Milestone “VV-L5-JA4B-14”
Submitted in Support of the FY05 ASC Level-2 Milestone #1357 “Code Verification”

François M. Hemez[1]

Los Alamos National Laboratory, X-Division (X-7)
Mail Stop F699, Los Alamos, New Mexico 87545

Abstract

This component of the FY05 level-2 milestone of the ASC “Code Verification” project investigates the formulation of non-linear error Ansatz equations for solution verification. The common practice in computational physics is to postulate that the error between the numerical solution of the discretized Partial Differential Equations (PDE) and the solution of the continuous equations only depends on the spatial discretization (or cell size Δh). The effect of time discretization (or time step Δt) is generally assumed to be a higher-order effect that can be neglected. The main goal of this work is to verify the extent to which such practice may be incorrect and, therefore, detrimental to the prediction of numerical solution uncertainty based on error Ansatz models parameterized with Δh alone. Limitations of the study are that it is based on: specific verification test problems for which the exact solutions of the continuous PDE are known (the more difficult case of solution self-convergence is not addressed); a single Eulerian code developed under the Los Alamos Code Project “Crestone” (other codes are not considered); uniform grids (AMR grids not considered); and constant time stepping (simulations are not analyzed with CFL-limited time steps like it is customary in computational physics). Four verification test problems are considered: Noh1D and Noh2D (convergent, shocked flow); Sedov1D and Sedov2D (divergent, shocked flow); Vortex2D (non-linear, smooth vortex evolution); and Wave2D (linear, smooth wave propagation). The analysis consists of solving each test problems nine times using different combinations $(\Delta h;\Delta t)$ of constant cell size and time step values. L^p-norms of the error e_h between numerical and exact solutions are then computed. Finally, statistical model fitting and analysis-of-variance techniques are deployed to analyze which effects such as $(\Delta h)^p$, $(\Delta t)^q$, or $(\Delta h)^r(\Delta t)^s$ best explain how the error e_h varies. Although results clearly depend on characteristics of the test problems (linear or nonlinear equations, convergent or divergent flow, smooth or shocked solution, 1D or 2D) and the state variables analyzed (density, energy, pressure, velocity), the overall conclusion is that the coupling between cell size Δh and time step Δt greatly matters to forecast numerical solution uncertainty. In addition to showing the importance that effects $(\Delta h)^p$, $(\Delta t)^q$, and $(\Delta h)^r(\Delta t)^s$ have on solution convergence, a general-purpose procedure based on the design of computer experiments, analysis-of-variance, and statistical model fitting is demonstrated for investigating non-linear error Ansatz models. Advantages of this approach are that it can be: applied in a somewhat black-box mode, that is, independently of the code or verification test problem; automated to a great extent; and extended to include non-constant cell sizes (to handle AMR refinement), non-constant time steps (to handle CFL-limited runs), and other parameters that control the discretization of PDE.

[1] Technical Staff Member in X-Division (X-7); Phone: 505-667-4631; Fax: 505-665-6722; E-mail: hemez@lanl.gov.

Table of Contents

(This document consists of 94 pages.)

Executive Summary

The main goal of solution verification is to develop a general-purpose capability to assess the convergence of numerical results. In computational physics, Partial Differential Equations (PDE) that govern the evolution of state variables (density, energy, pressure, velocity) are discretized for implementation and resolution on finite-digit computer arithmetic. The challenge is to verify that the approximate solutions of the discretized PDE converge to the exact solution of the continuous equations. In addition to assessing how "closely" the numerical solution matches the continuous solution, it is of great practical important to code users and analysts to forecast the solution error obtained with a given spatial discretization (or cell size Δh).

The dominant paradigm for solution verification postulates that a numerical solver provides an approximate solution $y_{h,t}$ that converges to the exact solution y^* of the continuous PDE at the convergence rate p. Such principle is written as:

$$e_h = \left\| y^* - y_{h,t} \right\| = \beta_1 (\Delta h)^p + \text{H.O.T.} \tag{1}$$

where e_h is the solution error defined as the norm ||•||, usually taken from the L^p, $W^{m,n}$, or H^p families, of the difference between the continuous and numerical solutions; Δh is a constant or characteristic cell size; and β_1 is a regression coefficient. Equation (1) is referred to as a ***linear error Ansatz model*** where the effect of time discretization is included in the Higher Order Terms (H.O.T.). Common practice is to obtain several numerical solutions $y_{h,t}$ on successively refined grids; compute the error norms $||y^*–y_{h,t}||$; and fit the unknown coefficients $(\beta_1;p)$ of equation (1) to verify that the rate of convergence (p) matches the theoretical value. The assumption that time discretization has no significant effect is justified by employing CFL-limited time stepping or performing multiple runs with the same (constant) time step Δt.

The main goal of this work is to verify the extent to which such practice may be incorrect and, therefore, detrimental to the prediction of numerical solution uncertainty based on linear error Ansatz models. Equation (1) is generalized to account for the non-linear coupling between cell size Δh and time step Δt:

$$e_h = \left\| y^* - y_{h,t} \right\| = \beta_1 (\Delta h)^p + \beta_2 (\Delta t)^q + \beta_3 (\Delta h)^r (\Delta t)^s + \text{H.O.T.} \tag{2}$$

Equation (2) is referred to as a ***non-linear error Ansatz model*** for solution verification. The overall same procedure is applied of obtaining several solutions $y_{h,t}$ by solving the discretized PDE multiple times with pairs (Δh;Δt) of successively refined cell sizes and time steps. Knowing the exact y^* and numerical $y_{h,t}$ solutions, the error e_h is computed for each run. The unknown coefficients $(\beta_1;\beta_2;\beta_3;p;q;r;s)$ are best-fitted to equation (2) and their values indicate the extent to which the coupling between cell size Δh and time step Δt matters to forecast the solution error.

Four code verification test problems for which closed-form, exact solutions y* of the continuous PDE have been published are considered for the analysis. Although relatively simple, these problems combine linear or non-linear wave propagation to smooth or discontinuous (shocked) solutions. The test problems solve the Euler equations of compressible, inviscid, and non-heat conducting gas dynamics. They are referred to as: Noh1D and Noh2D, a convergent, shocked flow problem; Sedov1D and Sedov2D, a divergent, shocked flow problem; Vortex2D, a non-linear, smooth vortex evolution; and Wave2D, a linear, smooth wave propagation.

Each test problem is analyzed with three constant cell size values $\Delta h = (\Delta h_C \mid \Delta h_M \mid \Delta h_F)$ where Δh_C, Δh_M, and Δh_F denote the coarse, medium, and fine refinement levels. Likewise, three constant time step values $\Delta t = (\Delta t_C \mid \Delta t_M \mid \Delta t_F)$ are defined. This choice results in a full-factorial design of computer experiments with (2 variables)$^{3\ \text{levels}}$ = 9 total runs. All runs are performed on the QSC machine at Los Alamos National Laboratory with the code RAGE (release 2005.03.31.000) and using constant cell sizes in all spatial directions (that is, no AMR refinement) and forced time stepping (that is, no CFL-limited runs). RAGE is a Eulerian hydro-dynamics code developed under the Los Alamos Code Project "Crestone". Figure 1 illustrates the errors e_h of the density, energy, pressure, and velocity solutions for the four test problems.

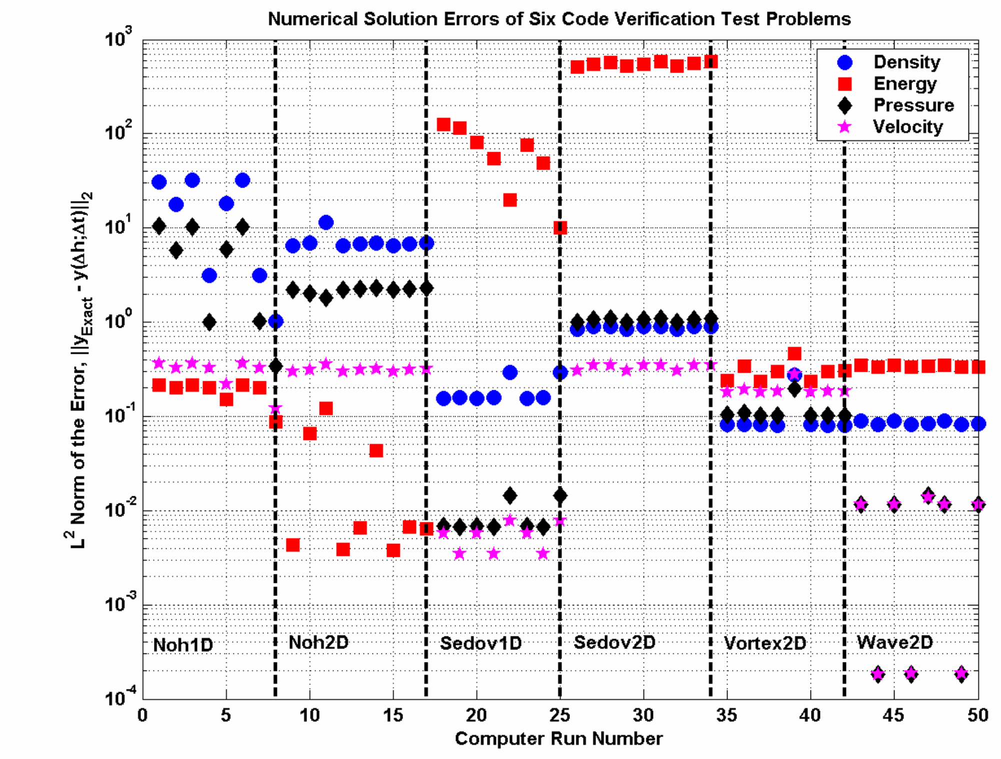


Figure 1. Solution errors obtained for four verification test problems.
Figure 1 shows the L^2 norms of the differences between exact and numerical solutions for four verification test problems. The dashed blue vertical lines separate the test problems, from left to right: Noh1D; Noh2D; Sedov1D; Sedov2D; Vortex2D; and Wave2D. The symbols represent different state variable solutions: density (blue circle); energy (pink square); pressure (red diamond); and velocity (black star). Runs performed on the QSC machine using the hydro-code RAGE (version 2005.03.31.000).

The analysis consists of fitting equation (2) to the data sets illustrated in Figure 1 for each test problem. Previously published statistical model fitting and ANalysis Of VAriance (ANOVA) techniques are used to analyze which effects $(\Delta h)^p$, $(\Delta t)^q$, or $(\Delta h)^r(\Delta t)^s$ best explain how the error e_h varies. Basically, this approach consists of answering questions such as: *"What controls the variability of errors e_h observed in Figure 1? Is the effect $(\Delta h)^p$ of cell size the most significant one? Is the effect $(\Delta t)^q$ of time step also significant?"* Beyond the assessment of which effects most matter to forecast the solution error, the statistical model fitting procedure also identifies the functional form of equation (2) that best-fits the data sets, its coefficients, and their posterior probabilities in a Bayesian sense.

A high-level illustration of the results is provided in Figure 2. Errors e_h obtained for the four test problems are analyzed simultaneously using a first-order ANOVA. The four effects considered are: 1) cell size Δh; 2) time step Δt; 3) problem type "id"; and 4) a variable "dmy" initialized at random. Because their influence factors are, on one hand, greater than those of variable "dmy" and, on the other hand, similar in magnitude to those of variable "id", Figure 2 demonstrates that both cell size and time step are significant to forecast the solution error. It is emphasized that the results illustrated in Figure 2 do not provide useful information to understand which error Ansatz model is most appropriate for a particular test problem. The test problems are analyzed individually in the report and results are shown to depend not only on their characteristics (1D vs. 2D, convergent vs. divergent vs. periodic flows, linear vs. non-linear equations, smooth vs. shocked solutions, etc.) but also on the state variable solutions (density, energy, pressure, or velocity).

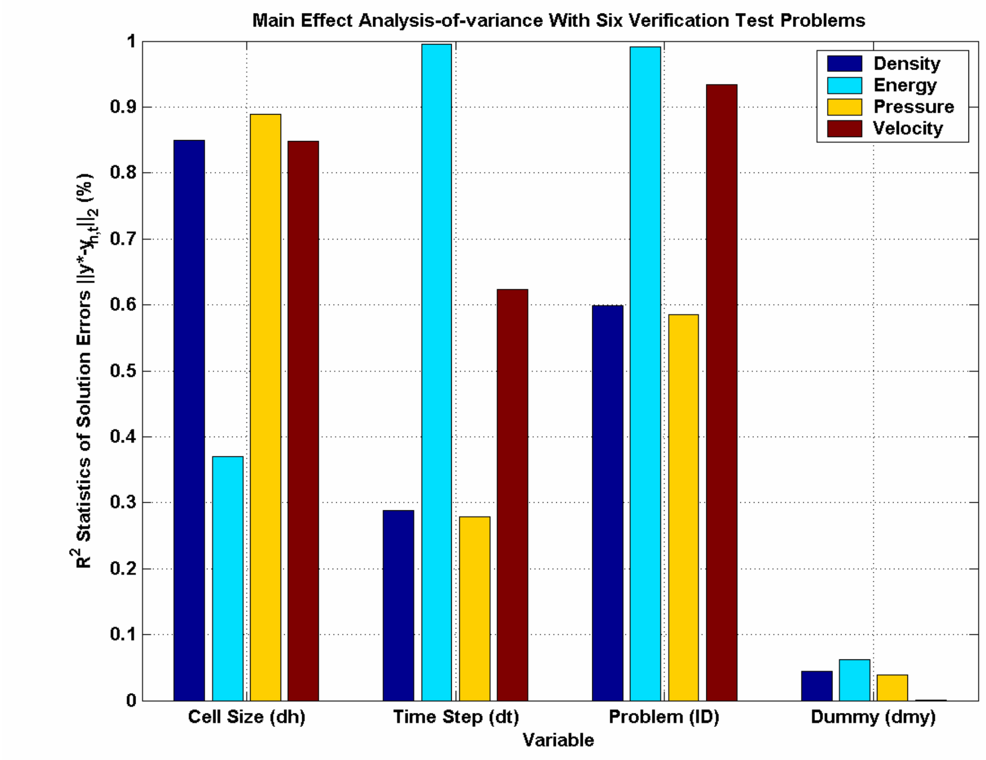


Figure 2. Results of an analysis-of-variance with four verification test problems.
Figure 2 illustrates an analysis-of-variance of the data sets shown in Figure 1. The R^2 statistic estimates Pearson's coefficient of correlation between values of a given factor (such as Δh, Δt) and values of the error e_h. A large R^2 indicates that the corresponding factor is important to explain how the error varies. The colors represent different state variable solutions: density (dark blue); energy (light blue); pressure (gold); and velocity (burgundy). The first two factors shown on the horizontal ordinate are cell size (Δh) and time step (Δt). The 3rd factor is the test problem (id) and the 4th factor is a “dummy” variable initialized at random (dmy). The ANOVA results show that both cell size Δh and time step Δh are significant to predict the value of the solution error e_h. Comparison with the significance levels of variables “id” and “dmy” provides two sanity checks: 1) the importance factors of Δh and Δt should be greater than those of “dmy” since the latter is a variable that has no effect on the solution error; 2) if they are indeed significant, then the importance factors of Δh and Δt should be similar to those of variable “id” because the latter represents the test problem as id = (Noh1D | Noh2D | Sedov1D | Sedov2D | Vortex2D | Wave2D).

The analysis is implemented through a dedicated software package called FEAST (Fitting Error Ansatz in Space and Time). FEAST is a collection of MATLAB™ functions currently specialized to the four aforementioned test problems. For a given test problem, FEAST provides a design of computer experiments (full factorial, two-level orthogonal array, Monte Carlo, or Latin Hypercube sampling); writes the corresponding input decks; and writes a macro-command to execute the individual runs on the QSC machine. Once the runs have been completed and transferred back from QSC, FEAST uploads the text dump files (obtained after post-processing the ASCII dump files of RAGE with AMHCTools); extracts the density, energy, pressure, velocity, and temperature solution fields; computes the error norms; and performs first-order

ANOVA, linear interaction ANOVA, deterministic model fitting, or statistical model fitting. About 5,000+ lines of codes have been written that can easily be extended to include other verification test problems, design of computer experiments, or analysis techniques.

The overall conclusion is that the coupling between cell size and time step greatly matters to forecast numerical solution uncertainty. In addition to showing the importance that effects $(\Delta h)^p$, $(\Delta t)^q$, and $(\Delta h)^r(\Delta t)^s$ have on solution convergence, a general-purpose procedure based on the design of computer experiments, analysis-of-variance, and statistical model fitting is demonstrated for investigating non-linear error Ansatz models. Advantages of this approach are that it can be: applied in a somewhat black-box mode, that is, independently of the code or verification test problem; automated to a great extent; and extended to include non-constant cell sizes (to handle AMR refinement), non-constant time steps (to handle CFL-limited runs), and other parameters that control the discretization of PDE.

Limitations of the study are that it is based on: specific verification test problems for which the exact solutions of the continuous PDE are known (the more difficult case of solution self-convergence is not addressed); a single Eulerian code developed under the Los Alamos Code Project “Crestone” (other codes are not considered); uniform grids (AMR grids are not considered); and constant time stepping (simulations are not analyzed with CFL-limited time steps like it is customary in computational physics). These limitations will be addressed in future work.

1. Introduction

Summary: The dominant paradigm for solution verification postulates that solvers provide approximate solutions that converge monotonically to the solution of the continuous equations. Conventional error Ansatz models for solution verification, therefore, account for linear and space-only discretization effects. Abnormal behaviors, such as oscillation or stagnation of numerical solutions as grids are refined, seem to indicate that other effects contribute to the numerical error. This study proposes to account for time discretization and to include the potential non-linear coupling between space and time effects. Another contribution is to establish the usefulness of statistical analysis techniques, such as the design of computer experiments, analysis of variance, and statistical model fitting, for solution verification. Limitations imposed to restrict the scope of the study are mentioned. They include focusing the study on the development of error Ansatz models; neglecting time-varying effects in the model parameters; and applying a single Eulerian hydro-code. Six objectives suggested to serve as evaluation criteria for completion of this level-5 component of the Code Verification project are listed. Finally, the contribution of this study to the broader context of quantification of numerical solution uncertainty is discussed.

Solution verification aims at developing a general-purpose capability to assess the convergence of numerical solutions. In computational physics and engineering, Partial Differential Equations (PDE) that govern the evolution of state variables are discretized for implementation and resolution on finite-digit computer arithmetic. The challenge is to verify that the approximate solutions of the discretized equations converge to the exact solution of the continuous equations. In addition to assessing how "closely" the numerical solution matches the continuous solution, it is often of great importance to verify that the observed convergence rate matches the one that should theoretically be provided by the numerical solver.

Solution verification can therefore be defined as a ***scientifically rigorous and quantitative process for assessing the mathematical consistency between continuum and discrete variants of PDE used to represent a reality of interest*** [5]. In simple terms, verification involves the comparison between the numerical solutions obtained from a calculation and a reference solution for a number of code verification test problems. One of the difficulties of solution verification resides in the fact that the exact solution of the continuous equations can be obtained analytically only in a few special cases that either feature a simple geometry, smooth dynamics, linearized operators, or combinations of the above. Few verification test problems exist that offer closed-form solutions. When they do, their applicability to the range of physics that must be simulated and verified for practical applications is limited if existent at all.

In general, one talks of ***code verification*** when the solution of the continuous equations can be derived analytically and in closed form, hence, providing a reference to which numerical solutions are compared. If the system of equations or problem setup are complicated enough that its continuous solution cannot be derived, then one talks of ***solution verification*** or solution ***self-convergence*** because no reference is available. Note that closed-form solutions sometimes involve approximation techniques such as the spectral decomposition or truncated series expansion. Even though this is strictly speaking not a correct statement, it is contended that obtaining a high-accuracy approximation of the continuous solution, approximation whose degree of accuracy can be controlled explicitly, qualifies as code verification activity.[2] This example illustrates that the boundary between code verification and solution verification is not crisp. What is unambiguous is the fact that any verification activity for which the degree of accuracy of the exact solution of the continuous equations cannot be controlled explicitly falls into the realm of solution verification.

Table 1-1. Classification of code and solution verification activities.

Verification Activity	Math. Consistency	Error Prediction
Software quality assurance	N/A	N/A
Code debugging	N/A	N/A
Code-solution comparison	•	
Assessment of convergence rates	•	
Evaluate a discretization error model	•	
Estimate errors for given length scale		•
Estimate length scales for given error objective		•

One can alternatively categorize verification activities either as ***mathematical consistency*** or ***error/length-scale prediction***. Table 1-1 makes this point by assigning verification activities to one or the other. The bottom line is that it is important to develop a methodology for code and solution verification by which specific test problems (with or without analytical solutions) can be identified for specific applications; solution error models can be identified; and these models can be used in a predictive mode to estimate the error of a numerical simulation given parameters such as the scale and time lengths.

The introduction is organized in six parts. Section 1-1 discusses the background of solution verification and motivates this investigation. Section 1-2 briefly defines what the study consists of and its scope is discussed in further details in Section 1-3. The general technical objectives suggested to serve as criteria for the evaluation of this component of the milestone are given in Section 1-4. Section 1-5 attempts to define this study in the broader context of the quantification of numerical solution uncertainty. Finally, Section 1-6 explains how the report is organized.

1.1 Background and Motivation

The dominant and only general-purpose paradigm for solution verification postulates that the numerical solver provides an approximate solution $y_{h,t}$ that converges monotonically to the solution y* of the continuous equations at the convergence rate p such that:

$$\left\| y^* - y_{h,t} \right\| = \beta_1 (\Delta h)^p + H.O.T. \tag{1-1}$$

where ||•|| denotes a norm, usually taken from the L^p, $W^{m,n}$, or H^p families, of the difference between the continuous and numerical solutions, and Δh denotes a characteristic volume, cell size, or element length of the spatially discretized computational domain. Equation (1-1) is referred to as a ***linear error Ansatz model*** where the effect of time discretization is explicitly or implicitly included in the <u>H</u>igher <u>O</u>rder <u>T</u>erms (H.O.T.) [13]. Common practice is to obtain several numerical solutions $y_{h,t}$ on successively refined grids; compute the error norms $||y^*–y_{h,t}||$; and fit the unknown coefficients (β_1;p) of equation (1-1) to verify that the rate of convergence (p)

[2] An example is the case where the solution of the continuous equations is given by an infinite series expansion such as a Fourier or Bessel (or any other) expansion. Since an infinite number of terms cannot be computed, truncation is needed. Strictly speaking, the resulting solution is not the “exact,” continuous solution because it is corrupted by the truncation error. Truncation error, however, can be estimated and controlled by ensuring that enough terms are included in the expansion. We therefore make the claim that such solution procedure qualifies as code verification.

matches the theoretical value.[3] The assumption that time discretization has no significant effect is justified by employing CFL-limited or stability-limited time stepping or performing multiple runs with the same (constant) time step Δt that verifies the CFL or stability condition with every computational grid.

When the exact solution y* of the continuous equations cannot be obtained analytically, solving equation (1-1) for the regression coefficient (β_1) and convergence rate (p) must rely on a simplified version of the error Ansatz model, as well as several approximations $y_{h,t}$ obtained from calculations made with successively refined grids. A simplified version of equation (1-1) is generally postulated where the norm of the solution error $||y^*-y_{h,t}||$ is replaced by a point-wise difference between exact and numerical solutions:

$$y_{h,t} = y^* + \beta_1 (\Delta h)^p + H.O.T. \qquad (1\text{-}2)$$

The advantage that such simplified error Ansatz model offers over equation (1-1) is that the exact-but-unknown solution y* can be estimated using, for example, the method of Richardson extrapolation [22].

The approach to solution verification illustrated by equation (1-2) relies on three strong, yet somewhat unverified, assumptions. First, convergence of the solutions $y_{h,t}$ must be monotonic. It implies that the discretization variables, such as space Δh, time Δt, angle Δθ, or frequency Δv, are selected to yield numerical solutions in the asymptotic domain of convergence. Whether or not a particular choice of Δh, for example, provides "near-to-converged" solutions is generally not known a priori and can only be verified a posteriori at the risk of making equations (1-1) and (1-2) irrelevant. Second, the estimate $\hat{y}^*$ must be "sufficiently close" to the exact-but-unknown solution y* of the continuous equations, which cannot be verified unless the numerical solutions are, again, known to have asymptotically converged. Third, it is assumed that the higher order terms have no significant influence and can be neglected.

Recent and not-so-recent attempts at applying this paradigm to the verification of numerical calculation for thermal, structural, or hydro-dynamics problems have reported "oscillations" or "stagnation" of the solution as the grid size is reduced, $\Delta h \rightarrow 0$, which may indicate that some of the aforementioned assumptions are not satisfied. Oscillatory convergence occurs, for example, at shock, rarefaction, or discontinuous boundaries created by contact conditions, all of which are important for non-smooth problems. A key operational issue is that such observations negate the commonly accepted belief that ***"more accuracy results from more zones or cells"*** [25]. Other practical difficulties include how to meaningfully define a characteristic time step (Δt) or grid size (Δh) in conjunction with Adaptive Mesh Refinement (AMR) studies or implicit time integration where these discretization variables can vary during the analysis.

1.2 Brief Definition of the Study

This component of the fiscal year 2005 Code Verification project of the Advanced Scientific Computing (ASC) program at Los Alamos investigates the correctness of formulating error Ansatz models for solution verification such as illustrated by equations (1-1) and (1-2). It proposes to develop, if appropriate, alternative error Ansatz models that account for a potential non-linear coupling of space and time discretization errors.

To verify the extent to which the current state-of-the-practice in computational physics and engineering may be incorrect and, therefore, detrimental to the prediction of numerical solution

[3] See Reference [12]. This reference is an example of state-of-the-practice code verification study where the functional form of the solution convergence error Ansatz model is not questioned.

uncertainty, equation (1-1) is generalized to account for the non-linear coupling between cell size Δh and time step Δt:

$$e_h = \left\| y^* - y_{h,t} \right\| = \beta_1 (\Delta h)^p + \beta_2 (\Delta t)^q + \beta_3 (\Delta h)^r (\Delta t)^s + H.O.T. \qquad (1\text{-}3)$$

Equation (2) is referred to as a ***non-linear error Ansatz model*** for solution verification. The overall same procedure is applied of obtaining several solutions $y_{h,t}$ by solving the discretized PDE multiple times with pairs (Δh;Δt) of successively refined cell sizes and time steps. Knowing the exact y* and numerical $y_{h,t}$ solutions, the error e_h is computed for each run. The unknown coefficients ($\beta_1;\beta_2;\beta_3$;p;q;r;s) are best-fitted to equation (1-3) and their values indicate the extent to which the coupling between cell size and time step matters to forecast the solution error.

The analysis procedure consists of solving a code verification test problem several times using different combinations (Δh;Δt) of constant cell size and time step values. L^p-norms of the error e_h between numerical and exact solutions are computed. Then, statistical model fitting and analysis-of-variance techniques are deployed to analyze which effects such as $(\Delta h)^p$, $(\Delta t)^q$, or $(\Delta h)^r(\Delta t)^s$ best explain how the error e_h varies and which effects should be included in the error Ansatz equation.

Beyond investigating the need for non-linear error models and importance of time effects, another contribution of this work is to establish the usefulness of statistical analysis techniques, such as the design of computer experiments, analysis of variance, and statistical model fitting, for solution verification. Fitting space-time error Ansatz models for solution convergence is not a novel idea and the contribution of previously published work is recognized [14]. However, the analysis techniques used in this effort are consistent with the interpretation of solution verification being an exercise in the quantification of uncertainty that originates from numerical discretization and resolution with finite-digit arithmetic.

1.3 The Restricted Scope of the Study

One of the main challenges of developing a general-purpose methodology for analyzing solution convergence error is that the exact solution y* of the continuous PDE may be unknown. The fact is it will almost always be unknown for real-world applications that feature non-trivial geometries, coupled physics, and complicated boundary or initial conditions. It is emphasized that the effort reported in this report is a first step towards the development of a methodology, therefore, restricted to the particular case of solution verification. Although more general and, arguably, more useful in terms of practical applications, the case of solution self-convergence is not addressed here.

It is also noted that the Method of Manufactured Solutions (MMS) is available to construct exact solutions for arbitrary systems of continuous or discretized equations [23]. Application of the MMS, however, seems most appropriate to build exact solutions that can be back-propagated through the code with the purpose of identifying programming errors. The invasive nature of the MMS, that requires the modification of source codes to enable, for example, the implementation of arbitrarily general forcing functions, makes this method unsuitable to obtain exact solutions from black-box codes. It is therefore not intended to make use of the MMS to build exact solutions for this investigation, hence, restricting the code verification test problems used to simple ones. Constructing analytical or numerical solutions with the MMS or other techniques, such as symmetry group and Lie algebra [3], will be the focus of future work.

Other techniques that may be brought to bear to study the convergence of numerical solutions include error bound analysis and Modified Equation Analysis (MEA). For example, MEA can be employed to quantify the truncation error provided by numerical integration of a system of equations. It produces the actual equation that the computer is solving numerically, that is, the combination of original equation and local truncation error where "local truncation error" is defined as the truncation error obtained by advancing the solution by a single time step (as opposed to the global truncation error). Understanding the effect of the local truncation error is important because ***"the accuracy of a numerical method is related to how well the balance between the different physical processes is preserved in the time integration scheme"*** [16]. Approaches such as error bound analysis and the MEA are not considered in this work, not by lack of interest but in an attempt to keep the investigation focused on the analysis of solution convergence error Ansatz models. It is recognized that these approaches are good candidates for future research.

It is also recognized that the contribution to solution convergence error of various effects, such as spatial discretization $(\Delta h)^p$, temporal discretization $(\Delta t)^q$, or space-time coupling $(\Delta h)^r(\Delta t)^s$, may change with time during the numerical simulation. This is because, as various physical effects are exercised in different proportions, the dominant contributor to the overall numerical error is not expected to remain the same. The effect on convergence, for example, of time discretization early during a hydro-dynamic simulation may be more pronounced than its effect late in the simulation. Likewise, the mechanism by which solution error is generated and accumulated may vary. Inadequate spatial discretization at some instants of the simulation may be replaced as dominant source of solution error by truncation errors at other times.

These remarks imply that the regression coefficients β_k of equations (1-1) to (1-3), as well as exponents (p;q;r;s), may be a function of time and/or space. To keep the study practical, any time-dependence or space-dependence of regression coefficients and exponents of the error Ansatz model is assumed to remain negligible and the results discussed in the report are limited to time-invariant (not time step-invariant) and space-invariant polynomial error Ansatz models. Other components of the Code Verification project are investigating some of these issues with, for example, the calculation of spatially distributed "conventional" error Ansatz models [24] based on equation (1-2) and spatially distributed grid convergence indices for AMR grids [8].

As far as the physics are concerned, the focus of this investigation is on smooth and non-smooth (shocked) hydro-dynamics problems represented by the non-linear Euler equations of compressible, inviscid, and non-heat conducting gas dynamics. Because finding the "correct" error Ansatz model is problem-specific and code-dependent, emphasis is placed on developing a flexible, general-purpose capability that accommodates non-linear Ansatz models represented by equation (1-3) as well as the conventional state-of-the-practice captured in equation (1-1).

Four code verification test problems for which closed-form, exact solutions y* of the continuous PDE have been published are considered for the analysis. Although relatively simple, these problems combine linear or non-linear wave propagation to smooth or discontinuous (shocked) solutions. The first test problem is the Noh problem, a convergent and shocked flow. The second test problem is the Sedov problem, a divergent and shocked flow. The third test problem is a vortex evolution problem that features a non-linear but smooth solution. The fourth test problem is a wave propagation problem with linear and smooth solution.

For simplicity, the analysis is restricted to the particular case of solution verification where the solution of the continuous equations is known (solution self-convergence is not considered here); with a focus on hydro-dynamics problems analyzed with uniform grids (Δh = constant, no AMR refinement) and forced time stepping (Δt = constant, no CFL-limited run); and applied only to the Eulerian hydro-code RAGE that is a code product developed under the Los Alamos Code Project "Crestone".

1.4 General Technical Objectives

The description of general technical objectives is extracted from the fiscal year 2005 Code Verification project level-2 milestone plan [5]. The four objectives inserted below are suggested in Reference [5] to serve as evaluation criteria for completion of this level-5 component of the project. The four general technical objectives are the following ones:

1) ***Review the relevant solution verification literature and error Ansatz model alternatives.***
2) ***Demonstrate a general-purpose methodology for developing and analyzing error Ansatz models.***
3) ***Perform the analysis and interpretation for several hydro-dynamics problems to be defined, but that should cover the range of linear/non-linear and smooth/non-smooth applications.***
4) ***Write and submit an internal report and the corresponding presentation materials.***

In addition, the milestone plan suggests that progress toward the completion of steps 5 and 6 below be reported at the end of the fiscal year, that is, October 2005:

5) ***Develop and archive an error Ansatz analysis code in Fortran or explain what it would take to integrate non-Fortran code into an ASC (or other) testing harness.***
6) ***Write and submit a journal article.***

This report is submitted as evidence that objectives 1-4 above are completed. Objective 5 is completed for the most part, although software for error Ansatz modeling and analysis was developed within the programming environment of MATLAB^{TM} and not in the Fortran language. Progress towards the completion of objective 6 has been achieved by submitting a contribution to the 47th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials (SDM) Conference to be held in Newport, Rhode Island, on May 1-4, 2006 [11]. This contribution has been accepted in the technical program and scheduled in a specially developed session dedicated to the Verification and Validation (V&V) of computational models.

1.5 Where Does the Study Fit in the Long-term Context of Verification?

The approach proposed in this report for developing solution convergence error Ansatz models applies to numerical solutions of space-dependent and time-dependent PDE. It is not, however, intended to be restricted to this class of problems. The long-term vision to which this work makes a (small) contribution is articulated around two main concepts.

First, it is reasonable to forecast that, as testing harnesses are developed and automation of repetitive tasks is improved, one may eventually be able to specialize the error Ansatz model developed to study solution convergence to a given code, physics, algorithm, or output quantity. The current paradigm in computational physics and engineering is that "one fits all" where the functional form of a solution convergence error model is restricted to space-dependent effects. This approach may be reasonable for some problems, such as calculations performed on AMR grids where the time step is defined by the CFL condition, but plenty of evidence also indicates that it may not always work. The long-term objective is to develop, not a single, but a ***catalog of error Ansatz models*** and categorize them by code, physics, algorithm, and output quantity of interest (density, pressure, flow velocity, energy, temperature, etc.).

To accomplish this first objective, semi-automated techniques, such as those proposed in this study, to develop and analyze error Ansatz models for solution verification will have to be demonstrated.

The second objective around which a long-term vision for solution verification is articulated is to develop a rigorous method for ***predicting confidence bounds of the (unknown) solution of the continuous equations***, given time and length scales selected by the code user for a particular application. Only timid attempts have been made so far in computational physics to propose bounds of numerical solution uncertainty. These are based, for example, on ad-hoc grid convergence indices, and a rigorous connection to uncertainty bounds at given statistical confidence levels is not made [21]. Statistical analysis techniques, such as those proposed in this study, may be brought to bear to calculate rigorous uncertainty bounds in the general case of solution self-convergence where the exact solutions of continuous equations are unknown.

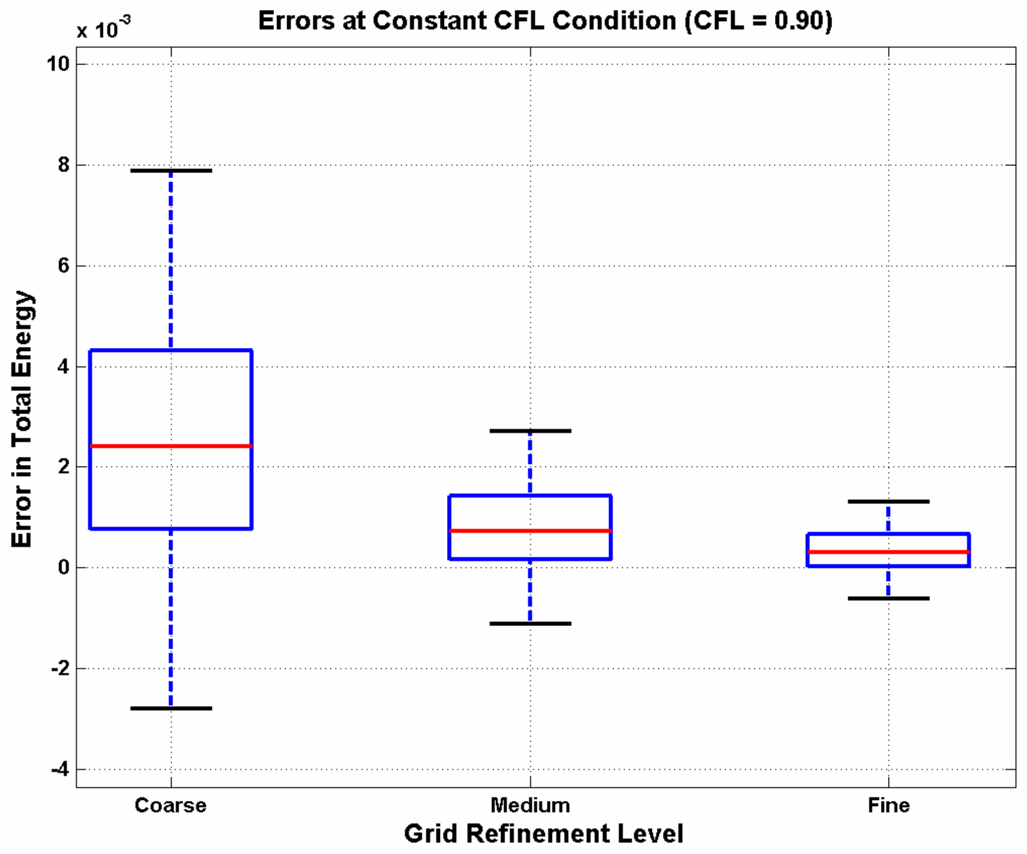


Figure 1-1. Illustration of an error Ansatz model and its lack-of-fit uncertainty bounds.

Figure 1-1 illustrates the concept of uncertainty quantification for solution convergence by showing predictions of a space-time Ansatz model similar to equation (1-3). The solution errors used to best-fit the parameters of equation (1-3) are defined as differences between exact and numerical solutions for the vortex evolution problem discussed in Sections 6 and 7. Computer runs are performed at various settings of (constant) cell sizes and CFL conditions. Uncertainty bounds shown as box-plots around the mean solution error (red solid lines) are obtained from a statistical boot-strapping technique, and they represent the lack-of-fit between data points and mean predictions of the error Ansatz model.

It is interesting to note that these uncertainty bounds tend to converge as the grid is refined. This observation suggests new approaches to assess solution convergence based on, not only the derivation of space-time error Ansatz equations, but also a description of how the lack-of-fit residuals ($||y^*-y_{h,t}|| - \hat{e}_h$) behave as a function of space and time discretization. Because the uncertainty bounds such as those shown in Figure 1-1 are obtained from statistical analysis, a rigorous connection to uncertainty bounds at given statistical confidence levels may be possible.

1.6 Organization of the Report

The report is organized in seven sections. Section 1 is the introduction that motivates the study and defines its scope and evaluation criteria. Section 2 introduces the main equations and notations used throughout the report. Typical issues of solution verification are discussed in Section 3 in non-technical terms. Section 4 is an investigation into the theoretical foundations of solution verification. Section 5 briefly summarizes the statistical approach proposed to screen the significant space or time effects and fit statistical error Ansatz models. Section 6 provides an illustration using the vortex evolution test problem and Section 7 discusses results obtained with the four code verification test problems.

The reader is made aware that the individual Sections that compose the report are, for the most part, independent from one another and can be read separately in non-sequential order. The reason for structuring the report in such a way is to provide easier access to the material presented, at the cost of duplicating some of it. The reader unfamiliar with solution verification should start with the introduction (Section 1) and mathematical formalism (Section 2). A subject-matter expert, on the other hand, can directly read details of the analysis approach proposed (Section 5) and numerical results (Sections 6 and 7).

2. Mathematical Formalism

Summary: Section 2 introduces the mathematical formalism used to investigate the convergence of numerical solutions. Discretized partial differential equations are based on the principle that the numerical solution converges to the exact-but-unknown solution of the continuous equations as the discretization is refined. In most practical cases, it means attempting to select "small enough" grid sizes and time steps. If the exact solution is known, convergence can be verified by fitting a polynomial equation where the error between exact and numerical solutions decreases monotonically as the grid size is reduced. Solution verification, in general, neglects the effect that coupling between space and time resolutions may have on convergence, or assumes that all calculations can be performed with a common, constant time step. Alternate error models are introduced that explicitly account for the non-linear coupling between space and time resolutions. The fact that the exact solution may not be known is also briefly addressed and the main equations of one approach known as the Richardson extrapolation are given. The purpose is simply to introduce the formalism and main equations of solution error verification. Underlying assumptions and resolution methods are discussed in the following Sections.

In this document a numerical model, or "code," that discretizes a set of coupled Partial Differential Equations (PDE) over a computational domain is denoted by:

$$y_{h,t} = M(\Delta h; \Delta t) \qquad (2\text{-}1)$$

where the independent variables discretized are denoted by the generic symbols h and t. The variables h usually represent the spatial discretization, whose characteristic size is denoted by Δh. Figure 2-1 shows examples of Δh, such as the size of a grid or angular discretization. Likewise, the symbol Δt in equation (2-1) represents a characteristic time step of the numerical simulation.

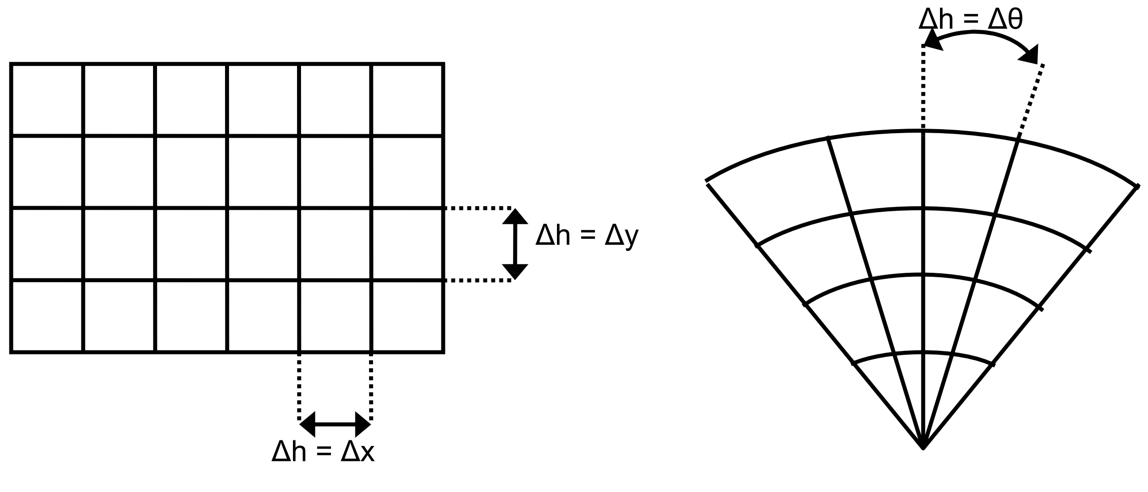


Figure 2-1. Illustration of grid or angular discretization sizes Δh.

The fact that neither Δh nor Δt may be constant during the numerical simulation is not relevant for the immediate discussion. This will be addressed later. What is important is that solvers based on the principle of discretization expect that the numerical approximation they calculate converges to the solution of the continuous equations as the discretization is refined:

$$\lim_{\Delta h \to 0, \Delta t \to 0} y_{h,t} = y^* \tag{2-2}$$

where y* denotes the exact solution of the continuous equations. Such formalism is common to all solvers based on discretization in computational physics and engineering, whether finite differences, finite elements, or finite volumes, among other approaches, are implemented.

Whether the exact solution y* is known from a closed-form, analytic expression or highly-accurate numerical solution or not has important implications. One talks of ***code verification*** when the objective is to verify that computations performed by the code are correct. Rigorously speaking, code verification requires that the exact solution be known for comparison with one or several numerical approximation(s) $y_{h,t}$. One talks of ***solution verification*** when the objective is to verify that the discretization provides asymptotically converged solutions. The terminologies "calculation verification" and "solution self-convergence" are also encountered in this case.

To assess the degree to which the numerical approximations have converged, one compares them to a reference solution. Clearly the best choice of reference solution is, again, the exact solution y* of the continuous equations. If the exact solution is unknown, then a numerical estimation obtained from solving the discretized PDE on the finest grid, or an extrapolation based on several grid calculations, can be used as reference solution. In this document the terminology "solution verification" is used somewhat indiscriminately but the context makes it clear whether or not the exact solution is known.

2.1 Linear and Non-linear Error Ansatz Models

The dominant and only general-purpose paradigm for solution verification postulates that the numerical solver provides an approximate solution $y_{h,t}$ that converges monotonically to the exact-but-unknown solution y* at a convergence rate p such that:

$$\left\| y^* - y_{h,t} \right\|_{L^p} = \beta_1 (\Delta h)^p + \text{H.O.T.} \tag{2-3}$$

where ||•|| denotes a L^p norm of the difference between the continuous and numerical solutions.[4] Clearly an error Ansatz[5] model such as equation (2-3) either assumes that the variable Δh is the only one whose discretization has any effect on the accuracy of the numerical approximation, or that the other discretization variables are kept constant. In the case of coupled PDE integrated in time over a computational grid, this implies that the time step Δt must be kept constant while satisfying the most stringent of <u>C</u>ourant-<u>F</u>riedrichs-<u>L</u>evy (CFL) conditions.[6]

[4] Note that the symbol p of the L^p norm has nothing to do with the symbol p that denotes the order of convergence of the numerical solver. For example, a quadratic convergence rate (or p = 2) can be assessed with L^1, L^2 or L^∞ norms. This notation should not lead to confusion because the terminology L^p is well recognized.

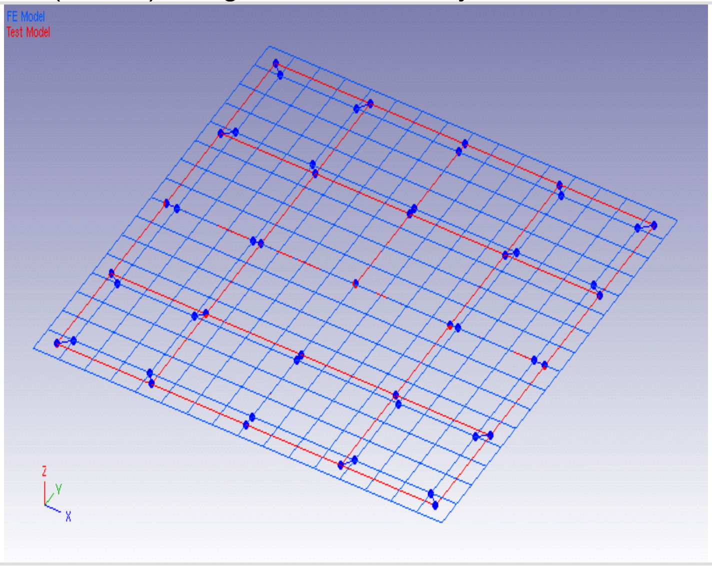
[5] The German word "der Ansatz" can be translated as "approach" or "estimate" in English, among other meanings. In the mathematical context it means "statement." An error Ansatz model refers to the equation(s) defined by an analyst to verify solution convergence.

[6] The CFL condition states that, in any time-marching computer simulation, the time step Δt must be less than the time for some significant action to occur. The CFL condition was originally formulated in the context of compressible fluid flows, where the time step must be kept smaller than the time it takes for a sound wave to propagate through the computational domain. Explicit time integration schemes introduce similar constraints due to their conditional stability conditions, not because of physical constraints.

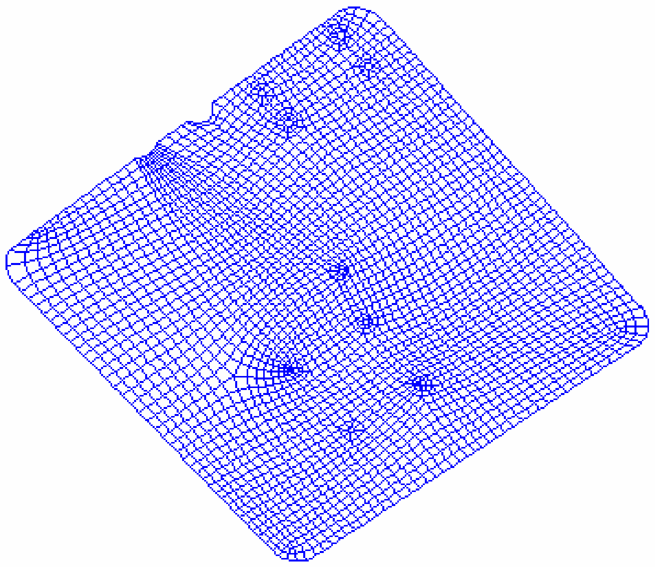
The approach proposed here for solution verification investigates other error Ansatz models such as non-linear models that introduce a coupling between the spatial discretization Δh and the temporal discretization Δt. In cases where it is suspected that space and time both influence convergence, or in practical situations where performing numerical simulations while varying Δh but keeping Δt constant is not possible, one may wish to verify solution convergence based on the generalized equation:

$$\left\| y^* - y_{h,t} \right\|_{L^p} = \beta_1 (\Delta h)^p + \beta_2 (\Delta t)^q + \beta_3 (\Delta h)^r (\Delta t)^s + \text{H.O.T.} \qquad (2\text{-}4)$$

where p and q denote the convergence rates in time and space, respectively; r and s are convergence rates for the coupling effect between time and space (that may or may not be relevant depending on the numerical solver implemented); and (β_1;β_2;β_3) are constant coefficients with little physical meaning other than providing sensitivity information. As before the Higher-Order Terms (H.O.T.) are ignored in the analysis.



(a) Coarse computational grid of a plate (element size Δh_C).



(b) Fine computational grid of a plate (element size Δh_F).

Figure 2-2. Coarse and fine finite element grids of a mesh convergence study.

Solving equation (2-3) or (2-4) for the coefficient(s) and convergence rate(s) is not possible unless several numerical approximations $y_{h,t}^{(1)}$, $y_{h,t}^{(2)}$, …, $y_{h,t}^{(N)}$ are obtained from calculations made with various combinations of grid sizes (Δh_1, Δh_2, …, Δh_m) and, possibly, time steps (Δt_1, Δt_2, …, Δt_n). Figure 2-2 illustrates two computational domains of a solid mechanics application: the coarse grid obtained with an element size Δh_C is refined to provide the finer grid whose characteristic element size is smaller, $\Delta h_F \leq \Delta h_C$. Four levels of mesh refinement are used in the convergence study of a finite element calculation.

Evaluating the error Ansatz model (2-3) without accounting for time discretization only requires two solutions $y_{h,t}^{(1)}$, $y_{h,t}^{(2)}$ obtained from calculations made with two grid sizes Δh_1, Δh_2 and a common time step Δt. Details of these calculations are provided in Section 4.

2.2 Convergence Verification With Extrapolated Solutions

Assessing the order of convergence and quantifying the solution error is difficult when the exact solution y* is not known analytically. Techniques based on the Richardson extrapolation have been proposed to replace the exact-but-unknown solution y* by an estimate ŷ from which the error Ansatz models can be written as:

$$\begin{aligned} \left\| \hat{y} - y_{h,t} \right\|_{L^p} &= \beta_1 (\Delta h)^p + \text{H.O.T.} \\ \left\| \hat{y} - y_{h,t} \right\|_{L^p} &= \beta_1 (\Delta h)^p + \beta_2 (\Delta t)^q + \beta_3 (\Delta h)^r (\Delta t)^s + \text{H.O.T.} \end{aligned} \quad (2\text{-}5)$$

Although equation (2-5) may appear similar to the error Ansatz models (2-3) and (2-4), a major difference is that the exact-but-unknown solution y* is replaced by an estimate ŷ. Taking this important step is valid only to the extent where the numerical solutions $y_{h,t}^{(1)}$, $y_{h,t}^{(2)}$, … $y_{h,t}^{(N)}$ have converged asymptotically. Of course asymptotic convergence cannot be guaranteed simply by choosing "small" grid sizes and time steps, nor can asymptotic convergence be verified a priori. This leads to a "catch-22" situation: to build an error Ansatz model an estimation of the exact-but-unknown solution is needed, which requires a sequence of converged numerical solutions. However asymptotic convergence cannot be assessed before analyzing the error Ansatz model.

Finally it is emphasized that there exists no closed-form solution to equations (2-5). The fact that an extrapolated solution ŷ is embedded in the calculation of the L^p norm of the error and the potential coupling term $(\Delta h)^r(\Delta t)^s$ between grid size and time step make it impossible to find an analytical solution in the general case. The only exception is when the prediction y is a scalar quantity: this simplifies the equations because the L^p norm reduces to an absolute value. The main equations of this special case are summarized below for completeness.

2.3 Solving the Error Ansatz Equations Explicitly

In the case of a scalar prediction, it is often assumed that the error Ansatz model is:

$$y_{h,t} = \hat{y} + \beta_1 (\Delta h)^p + \text{H.O.T.} \quad (2\text{-}6)$$

Because equation (2-6) features three unknown (ŷ; β_1; p), a minimum of three equations are needed. These are obtained by calculating numerical solutions using a coarse grid, a medium grid, and a fine grid. Neglecting any influence of the time integration and neglecting the higher-order terms leads to the following system of equations:

$$\begin{cases} y_C = \hat{y} + \beta_1 (\Delta h_C)^p \\ y_M = \hat{y} + \beta_1 (\Delta h_M)^p \\ y_F = \hat{y} + \beta_1 (\Delta h_F)^p \end{cases} \quad (2\text{-}7)$$

where y_C, y_M, and y_F denote numerical solutions obtained from the coarse mesh (grid size Δh_C), medium mesh (grid size Δh_M), and fine mesh (grid size Δh_F), respectively.

Note that this formalism implies that convergence occurs monotonically "from above" when $\beta_1 \geq 0$: the solutions obtained from several grids remain greater than the extrapolated solution, $y_C \geq \hat{y}$, $y_M \geq \hat{y}$, and $y_F \geq \hat{y}$. Monotonic convergence "from below" can be accommodated in a similar way with a negative regression coefficient, $\beta_1 \leq 0$. The formalism of equations (2-6) and (2-7), however, breaks down if convergence is oscillatory or if it stagnates as often observed in practice.

An illustration of non-monotonic convergence is provided in Figure 2-3. The grids shown in Figure 2-2, together with two other levels of refinement, are used to extract the 5th resonant frequency from the linear mass and stiffness matrices of a finite element representation of the plate.[7] Figure 2-3 illustrates the convergence of the frequency ω_5 as a function of mesh size Δh. Even though this calculation is linear, well-behaved, and the frequency ω_5 represents an integral quantity (ratio of internal energy to kinetic energy for the 5th mode shape), it can be observed that convergence is not monotonic. It is difficult to conclude if failure to match the theoretical convergence rate (p = 2 in this case because quadratic elements are used in the calculation) is due to non-converged solutions, inappropriate error Ansatz model, or a combination of both.

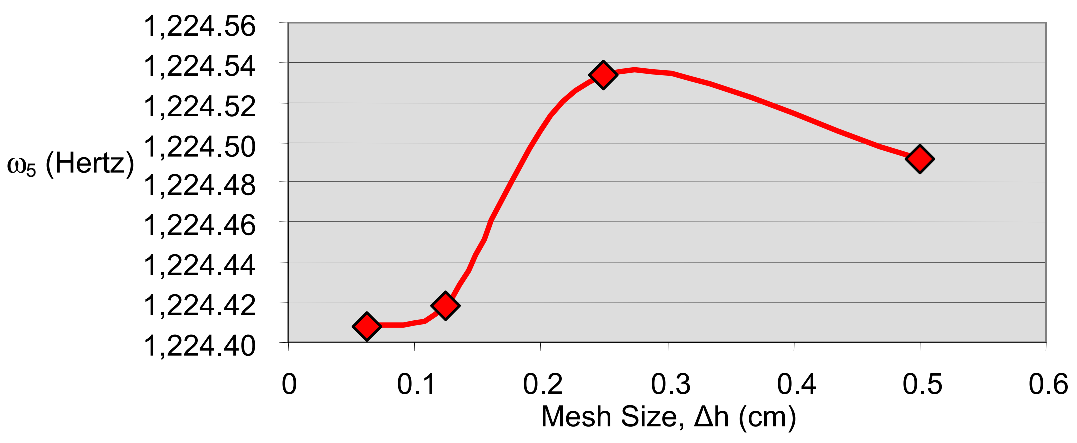


Figure 2-3. Frequency of the 5th resonant mode versus mesh size.

Under the aforementioned assumptions (scalar prediction; the convergence rate is affected by the grid size only; convergence is monotonic), the solution of the system of equations (2-7) is obtained analytically in the case of a constant refinement between the three grids:

$$p = \frac{\log\left(\dfrac{y_M - y_C}{y_F - y_M}\right)}{\log(R)} \qquad \text{(2-8)}$$

where R denotes the refinement ratio (R > 1) defined as $\Delta h_C = R\, \Delta h_M$ and $\Delta h_M = R\, \Delta h_F$. It is emphasized that a closed-form solution can be arrived at only if the refinement ratio is constant. The Richardson extrapolation of the exact-but-unknown solution is:

$$\hat{y} = y_F + \frac{y_F - y_M}{R^p - 1} \qquad \text{(2-9)}$$

[7] The mass and stiffness operators are discretized into matrices denoted M and K, respectively. The mode shapes ϕ_k and natural frequencies of vibration ω_k are the eigen-solutions of the spectral decomposition: $K\, \phi_k = \omega_k^2\, M\, \phi_k$.

where the order of convergence is either the theoretical value or the one calculated in equation (2-8). The regression coefficient β_1 can easily be back-calculated from equations (2-7) to (2-9).

2.4 Discussion

Verification activities are introduced and categorized broadly into code verification and solution (or calculation) verification. Roughly speaking the main difference between the two is whether the objective is to verify correctness of the code implementation or verify the degree to which the discretization provides an asymptotically converged solution.

A complicating factor of code and solution verification is the availability of the exact solution of the continuous PDE for comparison with the numerical approximation(s). Code verification requires that the exact solution be known explicitly. Comparing a numerical approximation to the exact solution provides assurance that the PDE are implemented, discretized, and solved correctly, therefore, verifying the code itself. Solution verification can be carried out whether or not the exact solution of the continuous equations is known. Availability of the exact solution makes it possible to rigorously verify the convergence rate or predict the numerical error associated with a given level of grid refinement. One talks of self-convergence study when the exact solution is unknown and replaced by an estimate defined from either the finest grid calculation or an extrapolation based on solutions obtained from several grid resolutions.

The discussion presented in this section briefly introduces the framework that has been proposed in the literature to verify the convergence of numerical solutions. The solution procedure outlined in equations (2-6) to (2-9) is commonly encountered in technical publications. It is nevertheless based on strong, yet, rarely discussed or verified, assumptions. Assumptions upon which the Richardson extrapolation of equation (2-9) is based are also questionable. These issues are raised in Section 3. The discussion stems from surveying unclassified studies of interest to the computational physics and engineering communities at Los Alamos. Section 4 derives the equations of code and solution verification and, while doing so, discusses the assumptions upon which they are based.

3. Short-comings of Solution Verification

Summary: Some of the short-comings of solution verification as it is commonly practiced are discussed in non-mathematical terms. The focus of the discussion is on issues that affect the development of error Ansatz models. Short-comings include, but are not restricted to, the seven following issues. 1) Monotonic polynomials cannot capture the sometimes oscillatory nature of solution convergence. 2) Space-only error models are inappropriate when numerical solutions stagnate as the grid is refined; 3) Higher-order terms of the error model are neglected, yet, they may contribute to the solution error. 4) The potential coupling between spatial and temporal discretization is not accounted for. 5) It is unclear how to define characteristic cell size and time step values when they vary during the calculation such as, for example, in AMR or CFL-limited time stepping runs. 6) The Richardson extrapolation employed to estimate the solution cannot be generalized easily to space-time error models. In cases where a weak formulation of the equations is solved, the Richardson extrapolated estimate and the solution of the continuous equations may be inconsistent because the two are based on different norms. 7) Finally, the role that error Ansatz models can play in the estimation of rigorous confidence bounds of the unknown solution needs to be explored for solution self-convergence.

This section is a short summary of issues identified with the "conventional" approach and state-of-the-practice for solution verification. These issues are discussed in simple terms, not mathematically, and the discussion is focused on problems encountered in the derivation of error Ansatz models for solution convergence. Section 4 provides more insight into the potential origin of some of these problems by examining the assumptions upon which the mathematical formalism for solution verification is built. Although the list below is not exhaustive, the claim is made that it captures some of the most important issues regarding the assessment of solution convergence based on space-only error Ansatz models such as equations (2-3) and (2-6).

1) Studies published in the literature and observations of solution convergence behavior often report that numerical solutions converge to the exact (known or not) solution of the continuous equations with oscillations as opposed to monotonically. Oscillations make it somewhat more difficult to assess whether or not the asymptotic range of convergence has been reached in the case of solution self-convergence where the solution of the continuous equations is unknown. Clearly, non-monotonic convergence is a situation that space-only error models such as those of equations (2-3) or (2-6) cannot handle.

2) Studies published in the literature also report that numerical solutions can "stagnate", that is, convergence to the exact (known or not) solution of the continuous equations becomes stationary as the computational grid is refined. This behavior suggests that effects other than the spatial discretization may influence solution convergence. This is also a situation that error Ansatz models such as those of equations (2-3) or (2-6) cannot handle since they are written as function of cell or element size only.

3) Conventional error Ansatz models may provide a poor goodness-of-fit to the data, that is, the polynomial model $e_h = \beta_1 (\Delta h)^p$ does not fit the observed solution errors with great fidelity. Lack-of-fit is generally detrimental to the prediction of solution error or length scale.

Simple solution convergence studies generally neglect (explicitly or implicitly) the effects of Higher-Order Terms (H.O.T.) in the polynomial. This may result in poor quality of fit to the data, as discussed in the previous bullet. H.O.T. are generally thought to include effects such as $(\Delta h)^r$ or $(\Delta h)^r(\Delta t)^s$ that approach zero faster than the main effect $(\Delta h)^p$ as both cell size (Δh) and time step (Δt) become vanishingly small. Implicit in the treatment of the H.O.T. is the assumption that the computational grid provides numerical values that are in the asymptotic regime whose definition is consistent with the error Ansatz model, so that the higher-order contributions can justifiably be neglected. Nevertheless, evidence is rarely presented that would validate the correctness of such assumption.

4) Space-only error Ansatz models such as those of equations (2-3) or (2-6) do not account for a potential coupling between spatial and temporal discretization or, in the case of solution self-similarity, a potential dependence of solution error on the cell size-to-time step ratio (Δh/Δt). Time stepping effects are usually neglected, either knowingly or due to omission or negligence. The main two mechanisms to eliminate the influence of time are to either run all calculations with time steps that satisfy the CFL or stability conditions, or employ the same (constant) time step for all calculations. The extent to which these commonly encountered practices guarantee independence of the solution convergence error from time effects and/or space-time coupling effects is generally not verified.

5) Defining characteristic cell size and time step values in cases where they vary during the numerical simulation is generally not addressed. Cell sizes change, for example, during an Adaptive Mesh Refinement (AMR) calculation that tends to concentrate a higher density of cells or zones where they are the most needed, that is, in areas of discontinuity or high gradient. Refinement such as h-refinement is another example that makes it difficult to define a characteristic element size during a calculation. Likewise, time stepping strategies can be implemented to adapt the choice of a time step to match an accuracy or stability criterion such as encountered in explicit calculations or in CFL-limited runs of hydro-dynamics or Computational Fluid Dynamics. It is unclear whether statistics such as the mean or standard deviation are pertinent choices, or if the maximum and minimum values should be used.

6) Is the common practice of defining a "reference" solution by extrapolating the results of several calculations, obtained from successively refined grids, appropriate to assess solution self-convergence when the solution of the continuous equations is unknown? This practice is referred to as the Richardson extrapolation when the error Ansatz model takes the form of equation (2-6). It is unclear whether Richardson extrapolation always provides an accurate estimation of the unknown solution. Furthermore, its generalization to other error Ansatz models, such as space-time models, is not straightforward.

When the resolution of the equations of motion or laws of conservation relies on a weak formulation, should solution self-convergence be assessed on the basis of an error norm that is consistent with the norm used to establish the equivalence between the strong and weak formulations? If so, the "reference" solution estimated from Richardson extrapolation may be inconsistent with the solution of the continuous equations because these two solutions are defined in the sense of different norms.

7) Finally, to what extent can error Ansatz models be used to estimate confidence bounds of the unknown solution of the continuous equations? The current state-of-the-practice in computational engineering is to rely on the Grid Convergence Index (GCI) to obtain uncertainty bounds but a rigorous connection to confidence bounds and statistical confidence levels has not, to the best of the author's knowledge, been demonstrated.

4. Conventional Approach to Solution Verification

Summary: What is referred to in this document as the conventional approach to solution verification is presented. Conventional error Ansatz models are those that rely on one or several of the five assumptions: 1) Assuming monotonic convergence; 2) Neglecting the potential coupling between time and space discretizations; 3) Relying exclusively on linear equations; 4) Substituting a Richardson extrapolation to the exact-but-unknown solution; and 5) Using scalar-valued quantities in lieu of what should be L^p norms. The intent of this section is to explain where each assumption comes into play, how it simplifies the equations or solution procedure, and under which circumstance it may lead to an inappropriate error Ansatz model. The first situation discussed is the case where an exact solution of the continuous equations is available. The discussion then proceeds to the case where no exact solution is available to better illustrate the simplifications made in the derivation of the conventional, Richardson extrapolation-based error Ansatz model and its solution.

This section presents the conventional approach to solution verification. The discussion arrives at the same equations as those introduced briefly in Section 2, the main difference being that the underlying assumptions are emphasized in a systematic way. The intent is to illustrate when the conventional approach to solution or self-convergence verification is appropriate and under which conditions some of the assumptions cease to be relevant.

Several variants to solution verification have been published and the discussion attempts to present them under a common logical framework. To explore all situations a progression is offered from the case where an exact solution of the continuous equations is available to the case where the exact solution is unknown and must be approximated by other means.

The starting point of solution verification resides in the principle of discretization of strong or weak formulations discussed in Section 4.1. The analysis of error Ansatz models where an exact solution of the continuous equations is available is presented in Section 4.2. Section 4.3 generalizes the discussion to the case where the exact solution is not available and must be replaced by an extrapolation. A solution procedure based on the Richardson extrapolation is derived in Section 4.4 and its underlying assumptions are discussed. Section 4.5 focuses on a standardized metric to report solution error convergence known as the Grid Convergence Index (GCI). Several misconceptions about the GCI are unraveled. Section 4.6 extends the closed-form solution obtained for solution verification under the assumption of monotonic convergence to the case of non-monotonic convergence. Finally a summary of the assumptions encountered throughout this discussion is given in Section 4.7. It is important to reach a clear understanding of these assumptions before starting to challenge some of them in Section 5 with a statistical approach to derive non-linear error Ansatz models.

4.1 Developing Solution Error From the Principle of Discretization

As mentioned in Section 2, the work discussed in this document applies to numerical simulations or "codes" that discretize coupled Partial Differential Equations (PDE) over a computational domain to make predictions represented symbolically by equation (2-1). For all practical purposes, the main independent variables discretized are space and time. The grid size and time step are denoted by Δh and Δt, respectively. It is emphasized that this notation does not necessarily imply that grid sizes and time steps are constant during the numerical simulation. This will be addressed in the sections that present numerical application results.

Discretization in space and time (and, possibly, other independent variables) is necessary because coupled PDE must be solved on the finite-digit arithmetic of computers and no closed-form solution exists in the case of non-trivial geometry, initial conditions, or boundary conditions. Numerical methods are based on the principle that the approximation they calculate converges to the solution of the continuous equations as the discretization is refined. This is expressed in equation (2-2), repeated here for convenience, as:

$$\lim_{\Delta h \to 0, \Delta t \to 0} y_{h,t} = y^* \tag{4-1}$$

Examples of modeling approaches based on the principle of discretization in computational physics and engineering include finite differences, finite elements, and finite volumes.

Finding a numerical solution is difficult when the PDE that govern the evolution of the solution dictate a high degree of regularity. For example an equation that features a second-order derivative requires a solution that is at least C^2 differentiable, which may be difficult to construct. Some of the modeling approaches previously mentioned, such as the finite element and finite volume methods, have in common that the search for a "strong" solution is replaced by the search for a "weak" solution. A strong solution is one that satisfies the equations of equilibrium or laws of conservation everywhere and at every instant. A weak solution, on the other hand, is one that satisfies the equations of equilibrium or laws of conservation in an average sense. The verification of the laws of conservation everywhere in the computational domain is relaxed, and it is somewhat easier to construct a numerical solution because its degree of regularity is lowered compared to that of the strong solution.

Mathematically speaking the strong solution y* that satisfies the continuous equations at every point and every instant solves a generic system of equations such as:

$$\begin{cases} A(y^*) = f, & \forall x \in \Omega, \quad \forall t \in [0;T] \\ y^* = y_o, & \text{at} \quad t = 0 \\ y^* = \partial y, & \forall x \in \partial\Omega \end{cases} \tag{4-2}$$

where A(•) denotes the mathematical operator that represents the continuous PDE; f denotes the external forcing function; Ω is the computational domain of boundary ∂Ω; [0; T] is the time period of interest; and y_o and ∂y are the initial and boundary conditions, respectively. The first one of equations (4-2) represents the equations of equilibrium or laws of conservations satisfied strongly in the entire domain and at every instant. The second and third equations represent the initial and boundary conditions. One constructs a weak solution $y_{h,t}$ that is equivalent to the strong formulation (4-2) by discretizing and solving the system of equations that corresponds to the weak formulation:

$$a(y_{h,t}; z_{h,t}) = l(f_{h,t}; z_{h,t}), \quad \forall z_{h,t} \in Z_h \subset Z, \quad \forall x_h \in \Omega_h \subset \Omega, \quad \forall t \in [0;T] \tag{4-3}$$

where a(•;•) represents the "weak" functional form of the operator A(•); l(•;•) is the inner product that accounts for the forcing function and boundary terms; and where the test functions $z_{h,t}$ are discretized from the appropriate space $Z_h \subset Z$. Defining the mathematical spaces to which the weak solution $y_{h,t}$ and test functions $z_{h,t}$ belong depends on the regularity of the operator A(•), type of boundary conditions, etc. Examples include the Banach spaces L^p and Sobolev spaces $W^{m,p}$ and $H^m = W^{m,2}$.

This brief discussion about how numerical solutions are constructed is included because it is our opinion that it is important, when defining an error Ansatz model, to keep in mind how the numerical solution is constructed. With weak solutions it is natural to postulate that a generic class of error Ansatz models is given by the equation:

$$e_h = \left\| y^* - y_{h,t} \right\|_{L^p} = \beta_1 (\Delta h)^p + H.O.T. \qquad (4\text{-}4)$$

where y* denotes the exact solution of the continuous equations. The order of convergence of the numerical algorithm is denoted by the symbol p. The symbol β_1 is a regression coefficient that has little practical interest other than indicating the sensitivity of the discretization error to the grid size Δh.[8]

The L^p norm is adopted for simplicity and because L^p spaces are the most general functional spaces for all practical purposes. ***The key point is that, because $y_{h,t}$ represents the discretized solution of weakly enforced laws of conservation, convergence must be studied in a norm consistent with the one used to establish that the strong and weak formulations are equivalent.*** The natural choice to prove the existence and uniqueness of weak solutions and construct error estimators is the so-called "energy norm" defined from the weak form of the operator, $||\bullet||^2 = a(\bullet;\bullet)$.[9,10] The only rigorous and defendable choice of norm to build an error Ansatz model would therefore appear to be the energy norm that is consistent with the derivation of the numerical solution.

This statement may seem like a severe restriction. In practice it is not because properties of usual functional spaces make it so that converging in the sense of a given norm implies that the solution also converges with other norms, although possibly at different rates. If a function belongs, for example, to the Sobolev space $W^{m,p}(\Omega)$, then it automatically belongs to $W^{k,p}(\Omega)$ for all $k \le m$. The inclusion "$W^{m,p}(\Omega) \subset W^{k,p}(\Omega)$" means that there exists a positive constant α such that $||\bullet||_{k,p} \le \alpha\, ||\bullet||_{m,p}$ where $||\bullet||_{k,p}$ and $||\bullet||_{m,p}$ denote the $W^{k,p}$ and $W^{m,p}$ norms, respectively, on the domain Ω. Establishing convergence in the sense of the norm $W^{m,p}$ therefore provides at least the same order of accuracy with the norm $W^{k,p}$. Likewise establishing convergence in terms of a L^p norm implies convergence in terms of any $W^{m,p}$ norm since "$L^p(\Omega) \subset W^{m,p}(\Omega)$," that is, there exists a positive constant α such that $||\bullet||_{m,p} \le \alpha\, ||\bullet||_p$ where $||\bullet||_{m,p}$ and $||\bullet||_p$ denote the $W^{m,p}$ and L^p norms, respectively.

Studying the convergence of discretized solutions, as opposed to functions that belong to infinite dimensional spaces such as discussed in the previous paragraph, is further facilitated by the fact that all norms defined over finite dimensional spaces are equivalent. It means that it is always possible to find two positive numbers μ_1 and μ_2 such that:

$$\mu_1 \left\| y_{h,t} \right\|_1 \le \left\| y_{h,t} \right\|_2 \le \mu_2 \left\| y_{h,t} \right\|_1, \quad \forall\, y_{h,t} \in H_h \qquad (4\text{-}5)$$

[8] Note that this error Ansatz model is appropriate to verify solution convergence as a function of spatial discretization only. Time discretization is excluded from the discussion for clarity, and will be addressed later.

[9] Several theorems have been proven that establish the existence and uniqueness of weak solutions for wide classes of problems. The work due to Lax-Milgram and Lions-Magenes, for example, can be cited as it applies to linear elliptic and linear parabolic PDE, respectively. (See the reference below.) The norms used to establish the equivalence between the strong and weak solutions are usually defined from $a(\bullet;\bullet)$ to take advantage of orthogonality properties when a numerical solution is constructed. A posteriori error estimators are also written with the energy norm $||\bullet||^2 = a(\bullet;\bullet)$. For example the generic form of error estimators developed for finite element analysis is:

$$\left\| y^* - y_{h,t} \right\| \le C_1 \left\| R \right\|_{L^2(\Omega)} + C_2 \left\| \partial r \right\|_{L^2(\partial\Omega)}$$

where R denotes an internal residual that accumulates the contributions of out-of-balance forces obtained inside each element; ∂r is the boundary residual that accumulates the contributions of tractions at element interfaces; and C_1 and C_2 are positive constants. Although the solution error can be bounded using such inequalities, it does not help to assess the order of accuracy as a function of element size Δh.

[10] See Reference [1].

where $||\bullet||_1$ and $||\bullet||_2$ are two norms defined over the finite-dimensional space $H_h \subset H$. The inequalities of equation (4-5) show that, if solution verification is studied with the norm $||\bullet||_1$, then the same order of convergence is obtained with the norm $||\bullet||_2$ as $\Delta h \to 0$ since:

$$\left\| y^* - y_{h,t} \right\|_1 = \beta_1 (\Delta h)^p + H.O.T. \tag{4-6}$$

implies that:

$$\mu_1 \left(\beta_1 (\Delta h)^p + H.O.T. \right) \le \left\| y^* - y_{h,t} \right\|_2 \le \mu_2 \left(\beta_1 (\Delta h)^p + H.O.T. \right) \tag{4-7}$$

Because the inequalities (4-7) are established for all solutions $y_{h,t} \in H_h$ and all grid sizes $\Delta h \ge 0$, it follows that:

$$\left\| y^* - y_{h,t} \right\|_2 = \beta_2 (\Delta h)^p + H.O.T. \tag{4-8}$$

for "sufficiently small" grid sizes and where the regression coefficient with the second norm (β_2) is different from the regression coefficient with the first norm (β_1). Comparing equations (4-6) and (4-8) concludes the proof.[11] ■

Error Ansatz models can therefore be constructed with any norm when numerical solutions result from weak formulations of the equations of equilibrium or laws of conservation. It should however be kept in mind that the only rigorous choice, in the sense of choosing a norm that it is consistent with the equivalence of weak and strong solutions, is the energy norm, $||\bullet||^2 = a(\bullet;\bullet)$, of the problem considered.[12]

One final point is made regarding the use of the generic error Ansatz model described by equation (4-9) and that can be found in numerous publications on solution verification:

$$y_{h,t} = y^* + \beta_1 (\Delta h)^p + H.O.T. \tag{4-9}$$

This error model appears similar to equation (4-4), the main difference being that the numerical solution is directly related to the exact solution of the continuous equations. Nonetheless the fact that no norm of the error is used is problematic for two reasons. First, equation (4-9) is true only to the extent where convergence is monotonic. It means that numerical solutions converge to the exact solution from above (that is, $y_{h,t} \ge y^*$) if the regression coefficient is positive, $\beta_1 > 0$. Likewise numerical solutions converge to the exact solution from below (that is, $y_{h,t} \le y^*$) if the regression coefficient is negative, $\beta_1 < 0$. Monotonic convergence may not happen in practice for a number of reasons, as illustrated in Section 2 and discussed in Section 3.

Second, the error Ansatz model (4-9) is defined with an error term ($y_{h,t}–y^*$), not a norm. It was emphasized previously that error Ansatz models can be written with any norm because all norms are equivalent over finite dimensional spaces, however, the error ($y_{h,t}–y^*$) is not a norm.

---

[11] The first implication of the proof is that convergence in the sense of a given norm $||\bullet||_1$ implies convergence in the sense of another one $||\bullet||_2$, both defined over a finite dimensional space H_h. The second implication stressed by equations (4-6) and (4-8) is that the convergence rate is the same with $||\bullet||_1$ and $||\bullet||_2$. It is emphasized that this result is true only for sufficiently small grid sizes, that is, to the extent where $\Delta h \to 0$. In practice different rates of solution accuracy may be observed depending on which norm is selected. For example it is often observed that convergence rates obtained with the L^2 and L^∞ norms are less than the L^1 convergence rate for shock propagation problems.

[12] The author realizes that this discussion is somewhat incomplete and the conclusion somewhat simplistic in the context of non-linear hyperbolic equations. This is because weak solutions for hyperbolic systems of equations are not necessarily unique and weak solutions may be found that are not, like in the case of linear elliptic equations, equivalent to the solution of the strong formulation. Other criteria, such as those which are entropy-based, need to be satisfied to eliminate the non-physical weak solutions. This results in adding significant numerical dissipation or implementing, for example, Total Variation Diminishing (TVD) integration schemes. Although these issues are not taken into consideration in the discussion, consistency between the norm used to assess solution convergence and the norm that defines the equivalence between weak and strong solutions nevertheless seems to be necessary even in the case of hyperbolic equations.

To the best of the author's understanding, there is no rigorous foundation for the derivation of the simplified error Ansatz model (4-9). It is convenient only because it is simple to understand and enables the derivation of a closed-form solution, as discussed in Section 4.4.

One may argue that the generic error Ansatz model (4-9) is the natural choice in the case of numerical solvers that seek strong solutions of the laws of conservation, as opposed to the solutions of weak formulations. An example of such methods is the finite difference approach. This actually depends on the type of numerical integration scheme. Some solutions obtained with finite differences do not satisfy exactly the equilibrium equations or laws of conservation at every single degree of freedom and instant.[13] There is, again, no justification for using equation (4-9) when this is the case. Extreme caution should therefore be exercised when using this error Ansatz model unless it is known a priori that it is appropriate.

4.2 Error Ansatz Models With Known Exact Solutions

So far it has been established that generic error Ansatz models for solution verification are equations that postulate that the solution error decreases monotonically as a function of the grid size Δh. The convergence of numerical solutions obtained from weak formulations of the equations of equilibrium or laws of conservation must be studied in the sense of the L^p norm (or, equivalently, the energy norm) consistent with the weak formulation:

$$e_h = \left\| y^* - y_{h,t} \right\|_{L^p} = \beta_1 (\Delta h)^p + \text{H.O.T.} \qquad (4\text{-}10)$$

The convergence of strong solutions of the equations of motion or laws of conservation can be studied in a point-wise sense:

$$e_h = \left| y^* - y_{h,t} \right| = \beta_1 (\Delta h)^p + \text{H.O.T.} \qquad (4\text{-}11)$$

In both cases the exact solution y* of the continuous equations is assumed to be available for calculating the solution error e_h.

In the general case where the order of convergence is unknown, these equations feature two unknowns: the order of convergence p and the regression coefficient β_1. Two equations are therefore needed to back-calculate the unknowns (p; β_1). Two such equations can be obtained by solving the same problem using two computational grids. The first grid is meshed with "large" cells using a characteristic grid or element size Δh_C, and it is therefore referred to as the coarse grid. The coarse grid provides a numerical approximation y_C. The second grid is meshed with "fine" cells using a characteristic grid or element size Δh_F, and it is referred to as the fine grid. The fine grid provides a numerical approximation y_F.[14]

As pointed out in Section 2, the error Ansatz models (4-10) and (4-11) assume that the grid size Δh is the only variable whose discretization has any effect on the accuracy of the numerical approximation, or that the time step Δt is kept constant for all numerical simulations. This implies that several numerical solutions must be obtained while keeping the time step constant and equal to the value that satisfies the most stringent of stability conditions, accuracy requirements, or Courant-Friedrichs-Levy (CFL) conditions. This may not be easy to achieve, especially in cases where implicit time integration schemes are implemented, Adaptive Mesh

[13] An example is a mid-point finite difference scheme (such as the trapezoidal rule) where the solution $y_{h,t}$ at time $(t+\Delta t)$ depends on solutions at previous time steps as well as a residual that assesses the lack of equilibrium at the half-point in time $(t+½\,\Delta T)$. The residual vector is often non-zero, which indicates that the solution variables satisfy the equations of equilibrium or laws of conservation only in an "average" sense.

[14] The symbol $y_{h,t}$ that denotes a numerical approximation of the solution y* of the continuous equations is simplified into y_C or y_F for the coarse-grid and fine-grid solutions, respectively.

Refinement (AMR) methods are used, or when equations of compressible flows are solved (where the speed of sound and CFL condition change).

Once the coarse-grid and fine-grid solutions have been computed, it is straightforward to establish that solution verification proceeds through the following steps:

$$\begin{aligned}&\text{1) Calculate the coarse-grid error:} \quad e_C = \left\| y^* - y_C \right\|_{L^p} \text{ or } \left| y^* - y_C \right| \\ &\text{2) Calculate the fine-grid error:} \quad e_F = \left\| y^* - y_F \right\|_{L^p} \text{ or } \left| y^* - y_F \right| \\ &\text{3) Solve the system of equations for p and } \beta_1: \quad \begin{cases} e_C = \beta_1 (\Delta h_C)^p \\ e_F = \beta_1 (\Delta h_F)^p \end{cases}\end{aligned} \tag{4-12}$$

The closed-form solution is given by:

$$p = \frac{\log\left(\frac{e_C}{e_F}\right)}{\log(R_{CF})} \tag{4-13}$$

where R_{CF} denotes the refinement ratio, always greater than one,[15] and defined as:

$$\Delta h_C = R_{CF} \Delta h_F \tag{4-14}$$

Note that this formalism does not prevent the analyst from using more than two grids to obtain a least-squares estimate of the order of convergence. This is however rarely done in practice due to the exponentially growing cost of subdividing an initial mesh into finer cells.

The procedure outlined in equation (4-12) relies on the assumption that the exact solution y* of the continuous equations is known. It must be either evaluated analytically (preferably without approximation or truncation of any kind) or programmed without having to rely on discretization of the computational domain. Finding closed-form solutions is generally possible only for problems that feature a simple geometry; simple initial and boundary conditions; and simplified physics where non-linearity, discontinuity, and coupling of various physical effects are neglected. Even though such test problems are of paramount importance for code verification, it is also true that their relevance to weapons physics is, at best, remote. The fact that an exact solution may not be available for even the simplest of test problems introduces the difficulty that the procedure (4-12) is not directly applicable in most practical cases.

4.3 Error Ansatz Models Without Exact Solutions

To address the case where an exact solution is not available, the solution error is defined as the L^p norm of the difference between a numerical approximation and a "best estimate" ŷ of the exact-but-unknown solution y*:

$$\hat{e}_h = \left\| \hat{y} - y_{h,t} \right\|_{L^p} = \beta_1 (\Delta h)^p + \text{H.O.T.} \tag{4-15}$$

The error Ansatz model (4-15) is consistent with numerical solutions obtained by solving a weak formulation of the laws of conservation. The norm that defines the solution error equation, shown to be the L^p norm for simplicity, should be equivalent to the norm used to prove the existence and uniqueness of the weak solution. Likewise the convergence of strong solutions of the equations of motion or laws of conservation can be studied in a point-wise sense with:

$$\hat{e}_h = \left| \hat{y} - y_{h,t} \right| = \beta_1 (\Delta h)^p + \text{H.O.T.} \tag{4-16}$$

[15] For example R_{CF} = 2 if the size of fine-grid cells is $(½)^D$ the size of coarse-grid cells, that is, each cell is subdivided in two in each direction in space (D=1, 2, or 3) to create a fine mesh from a coarse mesh.

In both cases the exact solution of the continuous equations is unknown and must be replaced by an estimate $\hat{y}$ to calculate the solution error $\hat{e}_h$ in equations (4-15) and (4-16).

Equations (4-15) and (4-16) state that the solution error decreases monotonically as the grid size is reduced ($\Delta h \to 0$), the main difference compared to the solution procedure (4-12) being that the true error is replaced by an approximation $\hat{e}_h$. The cost to pay is the addition of an additional unknown, $\hat{y}$. In theory three computational grids are sufficient to estimate the three unknowns (p; β_1; $\hat{y}$) of the coupled system of equations:

$$\begin{cases} \hat{e}_C = \beta_1 (\Delta h_C)^p \\ \hat{e}_M = \beta_1 (\Delta h_M)^p \\ \hat{e}_F = \beta_1 (\Delta h_F)^p \end{cases} \tag{4-17}$$

where the solution errors $\hat{e}_C$, $\hat{e}_M$, and $\hat{e}_F$ are obtained from three numerical solutions (y_C, y_M, and y_F) that solve the same system of PDE using coarse, medium, and fine grids, respectively. As before the notation is that these grids have characteristic cell sizes Δh_C, Δh_M, and Δh_F, and the influence of other discretization variables, such as the time step Δt, is assumed insignificant or has somehow been removed.

It can be observed that, besides the difficulty of obtaining multiple numerical solutions using finer grids, this approach is impractical whenever the solution $y_{h,t}$ is a field variable ($y_{h,t} \in \Re^N$ or belongs to another N-dimensional space Ω_h). Although a system of equations such as (4-17) could be solved through numerical optimization, it would be impossible to refine a computational grid (N+2) times for any realistic application. ***It is concluded that there exists no closed-form solution (p; β_1; $\hat{y}$) to the error Ansatz equations (4-15) or (4-16) when the exact solution is an unknown field (such as $y^* \in \Re^N$).***

One possibility for making this problem tractable would be to extrapolate the unknown solution, $\hat{y} \in \Re^N$ or $\hat{y} \in \Omega_h$, from the contributions of m basis functions ϕ_k where $m \le N$:

$$\hat{y} = \sum_{k=1 \ldots m} c_k \Phi_k \tag{4-18}$$

If the extrapolated solution can be parameterized with m generalized coordinates, then solving for the triplet (p; β_1; $\hat{y}$) only requires (m+2) independent equations because the unknowns of the error Ansatz model become (p; β_1; c_1; …; c_m). Clearly, the construction of a basis $\{\phi_1; \phi_2; \ldots; \phi_m\}$ is application dependent because the shape of the basis functions must be representative of the exact-but-unknown solution y*. A general procedure for obtaining such basis is through the <u>S</u>ingular <u>V</u>alue <u>D</u>ecomposition (SVD) of a data matrix, $Y_{h,t} \in \Re^N \times \Re^n$, that collects numerical solutions obtained from several grid resolutions:

$$Y_{h,t} = \begin{bmatrix} y_1(x_1) & y_2(x_1) & \cdots & y_n(x_1) \\ y_1(x_2) & y_2(x_2) & \cdots & y_n(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ y_1(x_N) & y_2(x_N) & \cdots & y_n(x_N) \end{bmatrix} = \sum_{k=1 \ldots n} \sigma_k (\Phi_k)(\Psi_k)^T \tag{4-19}$$

where $y_k(x_d)$ denotes the value of the kth numerical solution (k = 1 … n) at the dth grid point or cell (d = 1 … N). From this or another orthogonal decomposition of the numerical solutions, it may be possible to build a regression model to calculate the coordinates c_k of equation (4-18) as a function of grid size Δh, then, extrapolate their values for an asymptotically converged grid resolution ($\Delta h \to 0$). Of course such logic relies on the assumption that the shape functions, ϕ_k, provide a truncated basis upon which the exact solution can be accurately estimated.

To the best of the author's knowledge, such approaches have not been investigated and they currently remain a subject of open research. In any case it is doubtful that solutions can be obtained analytically because the derivation of a closed-form solution depends on whether the orthogonality properties of the basis $\{\phi_1; \phi_2; \ldots; \phi_m\}$ can be taken advantage of through the L^p norm used to define the error Ansatz model.

4.4 Closed-form Solution Procedure Using the Richardson Extrapolation

It is shown in Section 4.3 that there is no closed-form solution to the error Ansatz equations (4-15) or (4-16) in the general case where the exact solution of the continuous equations is not available. The only exception to this statement is the special case discussed here. When it is assumed that the prediction, y, is a scalar quantity and convergence is strictly monotonic, it is then possible to derive a closed-form solution based on the Richardson extrapolation.

The first assumption is that the response feature, y, is a scalar quantity. Examples include peak values (such as peak acceleration, pressure, or temperature) and integral quantities (such as statistical moments, lift and draft of the flow around an airfoil, fractal dimension of a turbulent flow). With a scalar prediction L^p norms simplify into an absolute value and the error Ansatz models (4-15) and (4-16) become similar. The main equation is repeated for convenience:

$$\hat{e}_h = \left|\hat{y} - y_{h,t}\right| = \beta_1 (\Delta h)^p + \text{H.O.T.} \tag{4-20}$$

The second assumption is monotonic convergence from "above" or "below." Monotonic convergence from above means that the numerical solutions are greater than the estimate of exact solution, or $y_{h,t} \geq \hat{y}$. Convergence from below means that the numerical solutions are smaller than the estimate of exact solution, or $y_{h,t} \leq \hat{y}$. Both cases are illustrated in Figure 4-1.

Clearly the only significant difference between convergence from above and convergence from below is the sign of the absolute value of the error in equation (4-20), which translates into the sign of the regression coefficient β_1. Without loss of generality, convergence from above is assumed in the remainder of this section which corresponds to a positive regression coefficient, $\beta_1 > 0$. It can be verified that the case of convergence from below leads to the same equations where the sign of coefficient β_1 simply needs to be switched from positive to negative.

Having assumed monotonic convergence from above, the absolute value can be simplified. The error Ansatz equations corresponding to the coarse, medium, and fine grids are:

$$y_C = \hat{y} + \beta_1 (\Delta h_C)^p, \quad y_M = \hat{y} + \beta_1 (\Delta h_M)^p, \quad y_F = \hat{y} + \beta_1 (\Delta h_F)^p \tag{4-21}$$

where $\beta_1 > 0$. The system of equations (4-21) must be solved for the triplet (p; β_1; $\hat{y}$). It is trivial to obtain the solution for the order of convergence by combining the above three equations to eliminate the other two unknowns, β_1 and $\hat{y}$. The value of the order of convergence is dictated by solving the non-linear equation:

$$p(\log(R_{MF})) + \log\left(1 - (R_{CM})^p\right) - \log\left(1 - (R_{MF})^p\right) = \log\left(\frac{y_M - y_C}{y_F - y_M}\right) \tag{4-22}$$

where R_{CM} denotes the refinement ratio from the coarse grid to the medium grid and R_{MF} is the refinement ratio from the medium grid to the fine grid. By definition the refinement ratios R_{CM} and R_{MF} are always greater than one, and related to the cell sizes through the formulae:

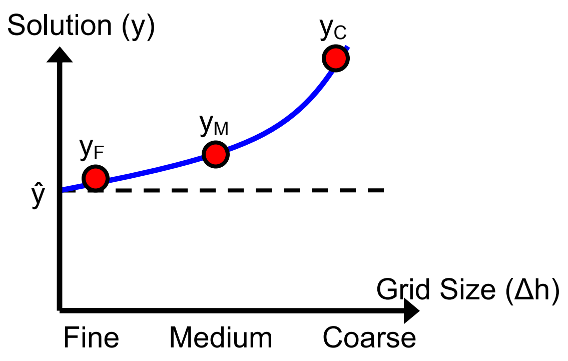
$$\Delta h_C = R_{CM} \Delta h_M, \qquad \Delta h_M = R_{MF} \Delta h_F, \qquad \Delta h_C = R_{CM} R_{MF} \Delta h_F \tag{4-23}$$

There exists no closed-form solution to the non-linear equation (4-22) when the refinement ratios R_{CM} and R_{MF} are different, and the value of p that satisfies the equation must be obtained through numerical optimization. However the equation further simplifies when the refinement ratios are equal, $R_{CM} = R_{MF}$, and it yields the well-known solution:

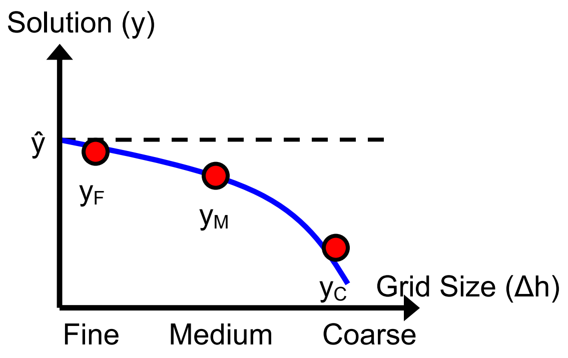
$$p = \frac{\log\left(\frac{y_M - y_C}{y_F - y_M}\right)}{\log(R)} \qquad (4\text{-}24)$$

where R denotes the common refinement ratio defined as:

$$R = \frac{\Delta h_C}{\Delta h_M} = \frac{\Delta h_M}{\Delta h_F} \qquad (4\text{-}25)$$



(a) Monotonic convergence "from above."



(b) Monotonic convergence "from below."

Figure 4-1. Illustration of monotonic solution convergence as a function Δh.

It is emphasized that the closed-form solution (4-24) for the order of convergence is based on three strong assumptions: ***1) The prediction y is a scalar quantity; 2) Convergence is strictly monotonic, either from above or below; and 3) The grid refinement ratio is constant as defined by equation (4-25).*** Clearly the third assumption of constant refinement ratio can be relaxed. The price to pay is that the order of convergence must then be computed from equation (4-22) using a numerical optimization solver. The second assumption of strictly monotonic convergence can also be relaxed, although this has never been discussed in the literature to the best of the author's knowledge. Section 4.6 presents a procedure to calculate the triplet (p; β_1; $\hat{y}$) based on equation (4-20) when the sign of the absolute value is unknown.

The strongest of the aforementioned three assumptions is clearly the first one. Although this is never discussed in the literature, it is contended that restricting solution convergence to scalar quantities so that simplified models such as $y_{h,t} = \hat{y} + \beta_1(\Delta h)^p$ can be studied is problematic. If an integral quantity is extracted from a field solution, then studying convergence based on a simplified error Ansatz model may be appropriate because an integral quantity "averages" the solution. Even though such averaging may not be equivalent to the L^p or energy norms of the weak formulation, it nevertheless implies that convergence is studied in an average sense and this is consistent with the logic of discretizing a weak formulation. On the other hand using a simplified error Ansatz model to assess convergence of a scalar quantity (such as a peak value) is not rigorous because the solution field converges in the sense of the L^p or energy norms while the error Ansatz equation $y_{h,t} = \hat{y} + \beta_1(\Delta h)^p$ studies convergence in a strong sense.[16]

Proceeding with the solution procedure, the extrapolation $\hat{y}$ is obtained as:

$$\hat{y} = y_F + \frac{y_F - y_M}{R^p - 1} \tag{4-26}$$

where the order of convergence is either the theoretical value or the estimation (4-24). Solution (4-26) is the Richardson extrapolation that best estimates the exact-but-unknown solution of the continuous equations. The solution shown in equation (4-26) is obtained from the medium and fine grids. It can be easily verified that using any combination of two grid solutions provides the same value for $\hat{y}$:

$$\begin{aligned} \hat{y} &= y_F + \frac{y_F - y_M}{R^p - 1} \\ \hat{y} &= y_M + \frac{y_M - y_C}{R^p - 1} \\ \hat{y} &= y_C + R^p\,\frac{y_M - y_C}{R^p - 1} = y_C + R^{2p}\,\frac{y_F - y_M}{R^p - 1} \end{aligned} \tag{4-27}$$

because the four estimates are not independent, they are related through equation (4-24). The regression coefficient β_1 can be back-calculated from any one of the following equations:

$$\beta_1 = \frac{y_C - y_M}{(\Delta h_M)^p\left(R^p - 1\right)} = \frac{y_M - y_F}{(\Delta h_F)^p\left(R^p - 1\right)} = \frac{y_C - y_F}{(\Delta h_F)^p\left(R^{2p} - 1\right)} \tag{4-28}$$

Equations (4-24), (4-26), and (4-28) calculate the triplet (p; β_1; $\hat{y}$) assuming that the prediction is a scalar quantity, convergence is strictly monotonic, and the grid refinement ratio is constant.

4.5 The Grid Convergence Index

The <u>G</u>rid <u>C</u>onvergence <u>I</u>ndex (GCI) was proposed by Patrick Roache in 1994 as a way to report the results of grid convergence studies in Computational Fluid Dynamics [21]. The GCI indicates convergence based on an error estimator derived from the Richardson extrapolation.

[16] An example is the calculation of the coefficients of lift C_L and drag C_D of an airfoil: they are integral quantities derived from the pressure field generated by a flow around an airfoil. In the case where a weak solution of the pressure field is obtained, assessing convergence of a scalar quantity such as the peak pressure with the error model $y_{h,t} = \hat{y} + \beta_1(\Delta h)^p$ is inconsistent with the fact that the pressure field satisfies the laws of conservation, and therefore converges, only in the sense of the L^p norm (or equivalent energy norm). Studying, on the other hand, the convergence of C_L and C_D based on the scalar equation may be appropriate because C_L and C_D provide an average of the pressure field.

The main advantage of the CGI is to provide an objective metric that uniformly reports the results of solution convergence no matter which order of accuracy and refinement strategy are used. In other words, the GCI value obtained from analyzing a 1st order method with grid "doubling" or "halving" can be directly compared to the GCI value obtained from analyzing a 2nd order method with non-constant mesh refinement.

The starting point of the GCI is an estimation of the exact-but-unknown solution ŷ provided through Richardson extrapolation, and re-written to explicitly show the error terms:

$$\hat{y} = y_F(1+e)$$
$$e = \frac{\varepsilon}{R^p - 1} \quad \text{and} \quad \varepsilon = \frac{y_F - y_C}{y_F} \tag{4-29}$$

where R denotes the grid refinement ratio and y_C and y_F are two numerical solutions obtained from calculations performed on coarse and fine grids, respectively. It is emphasized that equations (4-29) are based on two grids. This means that the order of convergence is known (and equal to the theoretical value or estimated independently from three computational grids) or the exact solution of the continuous equations is available (in which case $\hat{y} = y^*$).

The error (ε) defined in equation (4-29) is not the correct error estimator because it does not take into account the grid refinement ratio nor does it account for the order of convergence. The scaled error (e), on the other hand, takes the refinement ratio and order of convergence into account and forms the basis for the derivation of the GCI:

$$GCI = F_S \frac{|\varepsilon|}{R^p - 1} = \frac{F_S}{R^p - 1}\left|\frac{y_F - y_C}{y_F}\right| \tag{4-30}$$

where F_S is a safety factor whose value depends on the application, characteristics of the code used, and smoothness of the solution. Values $F_S = 1$, $F_S = 1.25$, or $F_S = 3$ have been proposed for various applications in Computational Fluid Dynamics and Solid Mechanics. Small values of the GCI, typically less than a few percents, provide strong evidence that the numerical solutions have converged to the exact-but-unknown solution y*, and that the Richardson extrapolation ŷ is an accurate estimation of y*.

One can understand that the GCI yields a uniform metric for comparing mesh convergence studies by noting that, for grid doubling or halving (R = 2) and 2nd order accuracy (p = 2), the denominator is $(R^p - 1) = 3$. By selecting a safety factor $F_S = 3$, the definition (4-30) reduces to $GCI = |\varepsilon|$. In cases where grid refinement is other than doubling ($R \neq 2$) or accuracy is other than quadratic ($p \neq 2$), the factor $F_S/(R^p - 1)$ effectively scales the error term $|\varepsilon|$ to provide a metric value that is comparable to one that would be obtained in the case of (R = 2; p = 2). It means that convergence studies can be compared and reported uniformly even when they are not conducted with grid doubling and 2nd order accurate methods.

As mentioned previously the derivation of the GCI in equations (4-29) and (4-30) is based on two computational grids which assumes that either the order of convergence is known or the exact solution of the continuous equations is available. The derivation can be generalized to calculate the GCI based on three grids in the case where the order of convergence and exact solution are unknown:

$$\hat{y} = y_F(1+e_F) \text{ and } GCI_F = F_S\frac{|\varepsilon_F|}{R^p - 1} \quad \text{where } e_F = \frac{\varepsilon_F}{R^p - 1} \text{ and } \varepsilon_F = \frac{y_F - y_M}{y_F}$$
$$\hat{y} = y_C(1+e_C) \text{ and } GCI_C = F_S\frac{|\varepsilon_C| R^p}{R^p - 1} \quad \text{where } e_C = \frac{\varepsilon_C R^p}{R^p - 1} \text{ and } \varepsilon_C = \frac{y_M - y_C}{y_C} \tag{4-31}$$

Symbols GCI_F and GCI_C denote values obtained from the fine-grid and coarse-grid solutions, respectively. Besides offering a generalization of the GCI to multiple grids, the key significance of definitions (4-31) is to provide a practical assessment of asymptotic convergence. It can be verified easily that asymptotic convergence has been achieved, that is, the numerical solutions y_C, y_M, y_F, and the extrapolation $\hat{y}$ are "close enough" to the exact-but-unknown solution y*, if:

$$\frac{GCI}{F_s(\Delta h)^p} \approx \text{Constant} \qquad (4\text{-}32)$$

which yields:

$$\frac{GCI_C}{GCI_F} \approx R^p \qquad (4\text{-}33)$$

The procedure outlined in equations (4-31) to (4-33) generalizes easily to multiple, that is, more than three, grids.

The attractiveness of the GCI resides in the fact that the safety factor (F_S) can be adjusted depending on the problem analyzed. It is shown above that the value $F_S = 3$ reduces the CGI to the relative error of two numerical solutions for grid doubling and 2nd order convergence. The common belief is also that the value $F_S = 3$ leads to a conservative estimate of the total numerical error that includes mesh/grid/zoning discretization error, Taylor series expansion error (if applicable), numerical integration error, and round-off. Although recommended by a large number of practitioners, there is no formal proof that the value $F_S = 3$ can be trusted to yield a conservative estimate of the numerical error.

More disturbing is the attempt to relate the error metric, |e| of definition (4-29) or $|e_F|$ in equations (4-31), to the concept of confidence interval. For example Patrick Roache writes:

> *"The error estimator e_F itself does not provide a very good confidence interval. One might expect that it is equally probable that e_F be optimistic as conservative, i.e. it is just as likely that the actual error be greater than e_F as less than e_F. This would correspond roughly to a 50% confidence bound."* (From Reference [22].)

Although the first sentence above stops short of saying that e_F defines a confidence interval, it engages on the slippery slope of suggesting a connection between the two. Unfortunately other authors do not exercise the same level of restraint and the belief that *"a fine-grid GCI computed with a factor of safety $F_S = 3$ defines a 95% confidence interval of the exact solution"* is often heard in oral presentations and read in technical publications. ***The idea that the error e_F and the GCI define a confidence interval of grid convergence uncertainty at the 95% or any other confidence level is simply wrong.*** Two arguments are presented in defense of this opinion. First, confidence intervals make sense only in the context of statistical distributions. To the best of the author's understanding, there is no connection between the GCI derived in equations (4-29) to (4-31) and a hypothetical distribution of the solution error.

Second, the common belief alluded to in the above quotation that a 50% confidence interval [y_{Min}; y_{Max}] is the interval in which there is a 50% chance of finding the true-but-unknown value of the solution y* is simply wrong. The concept of confidence interval applies only to a population statistic such as a mean μ_Y. The notation $P(y_{Min} \le \mu_Y \le y_{Max}) = 95\%$, for example, defines the interval [y_{Min}; y_{Max}] as the 95% confidence interval of the statistic μ_Y. This does ***not*** mean that there is a 95% chance that the true-but-unknown population mean is between y_{Min} and y_{Max}. It means instead:

> *"If a hundred samples of N data points were taken and the 95% confidence intervals were calculated for each sample, then 95 of these intervals would intersect the true population mean while 5 intervals would not contain the true population mean."* (From References [10] and [4].)

Likewise the belief according to which *"using $F_S = 3$ in the definition of the GCI provides 2-σ uncertainty bounds"* is often encountered. The terminology 2-σ refers to twice the standard deviation (σ) of the distribution of grid convergence uncertainty which, roughly, corresponds to the 95% probability level of a Gaussian distribution. It is undeniable that defining 2-σ bounds would be convenient to bound the exact-but-unknown solution y* relative to the estimates y_C, y_M, y_F, and extrapolation ŷ. Patrick Roaches, for example, discusses a 2-σ uncertainty band when analyzing a population of 30 solutions for the 1D Burgers equation solved with different combinations of grid refinement ratios, orders of accuracy, Reynolds numbers, etc. These 2-σ bounds are legitimate because they apply to a population of solutions [22]. Unfortunately many practitioners may have understood this to mean that "*$F_S = 3$ provides 2-σ uncertainty bounds"* which implies that the convergence uncertainty is such that $|y^* - y_F|$ is distributed according to a population whose mean statistic is zero and standard deviation statistic is $2\sigma = GCI_F$. ***The idea that the GCI defines 2-σ uncertainty bounds is wrong.*** A statistically correct statement made about a population of problems has no justification when reduced to a single problem.

4.6 Closed-form Solution For Non-monotonic Convergence

In equations (4-24), (4-26), and (4-28), a closed-form solution is derived for the triplet of unknowns (p; β_1; ŷ) from the knowledge of three numerical solutions y_C, y_M, y_F. The procedure is made possible only because scalar quantities are analyzed, convergence is monotonic, and the grid convergence ratio is constant. This section discusses a procedure to obtain (p; β_1; ŷ) based on equation (4-20) when the assumption of monotonic convergence is relaxed. Like before a closed form solution is possible only in the case of constant grid refinement. If the coarse-to-medium refinement is different from the medium-to-fine refinement, a solution is still possible although numerical solvers (such as a Newton-Raphson search) must be implemented to solve the non-linear equation.

The three equations needed to solve for the triplet (p; β_1; ŷ) are given by:

$$\begin{cases} |\hat{y} - y_C| = \beta_1 (\Delta h_C)^p \\ |\hat{y} - y_M| = \beta_1 (\Delta h_M)^p \\ |\hat{y} - y_F| = \beta_1 (\Delta h_F)^p \end{cases} \qquad (4\text{-}34)$$

where p denotes the order of convergence (p = 2, for example, for second-order accuracy), ŷ is the Richardson extrapolation of the exact-but-unknown solution y* of the continuous equations, and β_1 is a regression coefficient. Assuming monotonic convergence simplifies equations (4-34) where absolute values disappear and the sign of coefficient β_1 is constant, either $\beta_1 > 0$ for a convergence from above or $\beta_1 < 0$ for a convergence from below (see Figure 4-1).

Keeping the absolute values in the system of equations (4-20) recognizes that convergence may not be strictly monotonic. Figure 4-2 illustrates a non-monotonic convergence where the solution error changes its sign but consistently decreases as a function of grid size Δh. Even though the absolute value is not differentiable at the origin, solving the system of equations to calculate (p; β_1; ŷ) is tractable using a numerical optimization solver and three equations only.

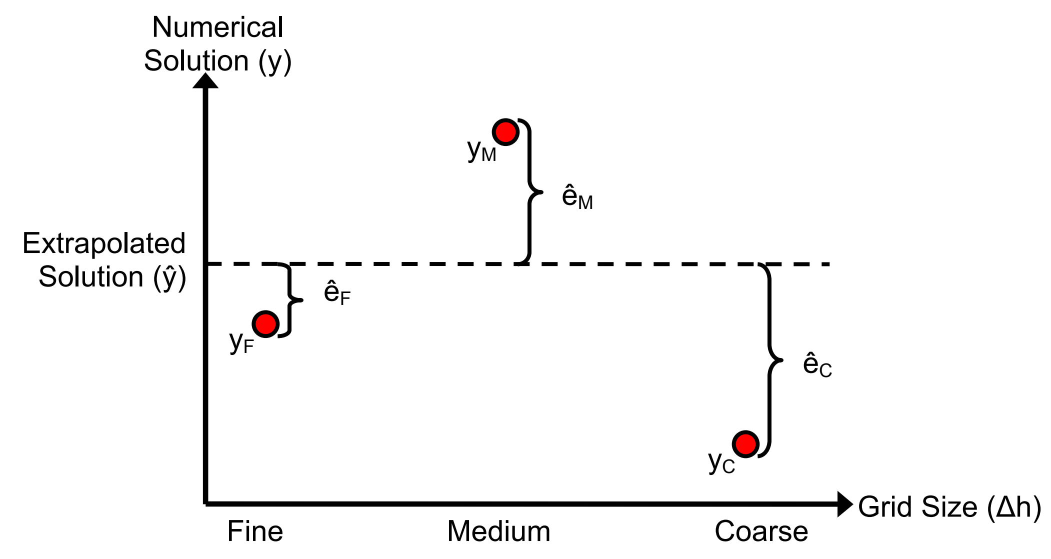


Figure 4-2. Illustration of non-monotonic solution convergence as a function Δh.

A solution procedure is proposed here that leads to a closed-form solution. To remove the absolute values from the system of equations (4-34), the signs s_C, s_M, and s_F of each difference are written explicitly:

$$\begin{cases} y_C = \hat{y} + s_C \beta_1 (\Delta h_C)^p \\ y_M = \hat{y} + s_M \beta_1 (\Delta h_M)^p \\ y_F = \hat{y} + s_F \beta_1 (\Delta h_F)^p \end{cases} \qquad (4\text{-}35)$$

where the regression coefficient β_1 is, now, positive only. The two systems of equations (4-34) and (4-35) are equivalent in terms of information content only if the signs s_C, s_M, and s_F are unknown. Since each sign is either positive or negative, the total number of combinations of the triplet (s_C; s_M; s_F) is equal to $(2)^3 = 8$. The eight cases are listed in Table 4-1.

If the error Ansatz model (4-20) with non-monotonic convergence is the correct assumption, then at least one of the hypotheses listed in Table 4-1 must be verified. A brute-force approach to solve for the triplet (p; β_1; $\hat{y}$) consists of solving the system of equations (4-35) for each combination of signs (s_C; s_M; s_F) listed in Table 4-1 and examining which solutions make sense.

Table 4-1. Listing of the eight possible cases for the triplet of signs (s_C; s_M; s_F).

Hypothesis	**Possible Cases**			**Signs of Absolute Errors**		
	Solution y_C	**Solution y_M**	**Solution y_F**	**Sign s_C**	**Sign s_M**	**Sign s_F**
(H_1)	$\hat{y} \geq y_C$	$\hat{y} \geq y_M$	$\hat{y} \geq y_F$	-1	-1	-1
(H_2)	$\hat{y} \geq y_C$	$\hat{y} \geq y_M$	$\hat{y} \leq y_F$	-1	-1	+1
(H_3)	$\hat{y} \geq y_C$	$\hat{y} \leq y_M$	$\hat{y} \geq y_F$	-1	+1	-1
(H_4)	$\hat{y} \geq y_C$	$\hat{y} \leq y_M$	$\hat{y} \leq y_F$	-1	+1	+1
(H_5) = -(H_4)	$\hat{y} \leq y_C$	$\hat{y} \geq y_M$	$\hat{y} \geq y_F$	+1	-1	-1
(H_6) = -(H_3)	$\hat{y} \leq y_C$	$\hat{y} \geq y_M$	$\hat{y} \leq y_F$	+1	-1	+1
(H_7) = -(H_2)	$\hat{y} \leq y_C$	$\hat{y} \leq y_M$	$\hat{y} \geq y_F$	+1	+1	-1
(H_8) = -(H_1)	$\hat{y} \leq y_C$	$\hat{y} \leq y_M$	$\hat{y} \leq y_F$	+1	+1	+1

A remark is that cases labeled (H_5) to (H_8) are "opposite" from cases labeled (H_1) to (H_4) in the sense that they feature combinations of signs that are switched. This symmetry is denoted symbolically as "(H_{4+k}) = -(H_{5-k})" in Table 4-1 with k = 1, 2, 3, 4. It means that solutions for cases (H_5) to (H_8) can be obtained through simple modifications of solutions (H_1) to (H_4), hence, giving four independent cases only. The number of independent cases actually reduces to three since hypothesis (H_1) and its opposite (H_8) correspond to the case of monotonic convergence and can be eliminated from the present analysis. The remainder of the discussion is focused on cases (H_2), (H_3), and (H_4) where one sign is different from the other two. The derivations below are nevertheless valid for the eight cases of Table 4-1.

The solution procedure is as follows. Combining the coarse and medium equations (4-35) to eliminate the unknown Richardson extrapolation leads to an expression for the regression coefficient β_1 as a function of the order of convergence p:

$$\beta_1 = \frac{y_C - y_M}{s_C(\Delta h_C)^p - s_M(\Delta h_M)^p} \tag{4-36}$$

The same derivation is applied to the coarse and fine solutions:

$$\beta_1 = \frac{y_C - y_F}{s_C(\Delta h_C)^p - s_F(\Delta h_F)^p} \tag{4-37}$$

and the medium and fine solutions:

$$\beta_1 = \frac{y_M - y_F}{s_M(\Delta h_M)^p - s_F(\Delta h_F)^p} \tag{4-38}$$

Any two of equations (4-36), (4-37), (4-38) can be combined to calculate the value of the order of convergence that leads to a unique solution for β_1. Combining, for example, the second and third leads to:

$$s_C(y_F - y_M)(\Delta h_C)^p + s_M(y_C - y_F)(\Delta h_M)^p + s_F(y_M - y_C)(\Delta h_F)^p = 0 \tag{4-39}$$

which can be expressed in terms of the refinement ratios R_{CM} and R_{MF} of definition (4-23):

$$s_C(y_F - y_M)(R_{CM}R_{MF})^p + s_M(y_C - y_F)(R_{MF})^p + s_F(y_M - y_C) = 0 \tag{4-40}$$

For any combination of signs (s_C; s_M; s_F) from Table 4-1, the non-linear equation (4-40) can be solved for the order of convergence using, for example, a Newton-Raphson search. Uniform refinement across the three grids, that is, $R_{CM} = R_{MF} = R$, provides a further simplification into:

$$s_C(y_F - y_M)R^{2p} + s_M(y_C - y_F)R^p + s_F(y_M - y_C) = 0 \tag{4-41}$$

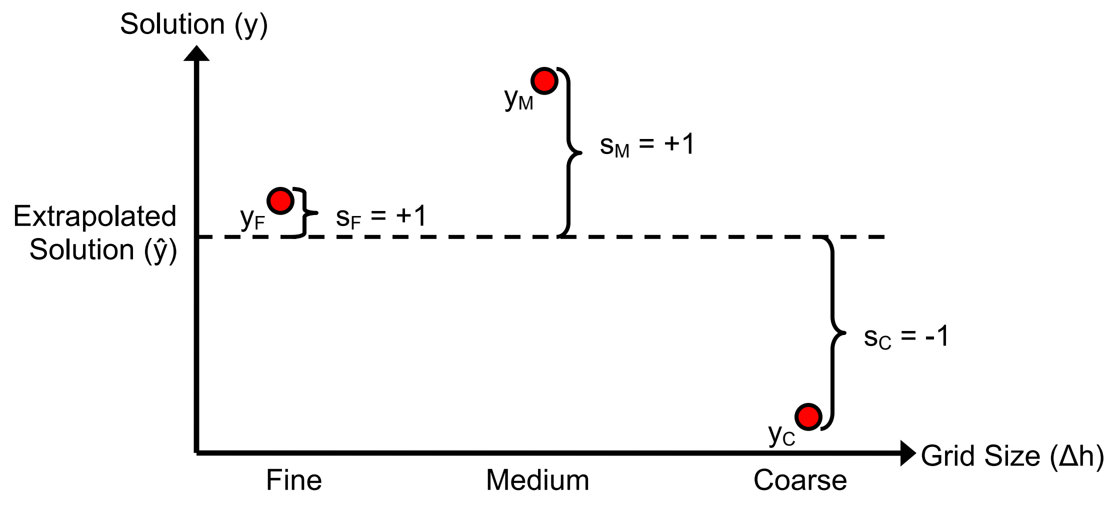
Equation (4-41) is made quadratic through the change of variable $X = (R)^p$, which leads to the solution procedure outlined below:

$$\begin{aligned} &\text{1. Solve the quadratic equation:} \quad s_C(y_F - y_M)X^2 + s_M(y_C - y_F)X + s_F(y_M - y_C) = 0 \\ &\text{2. Calculate the order of convergence:} \quad p = \frac{\log(X)}{\log(R)} \end{aligned} \tag{4-42}$$

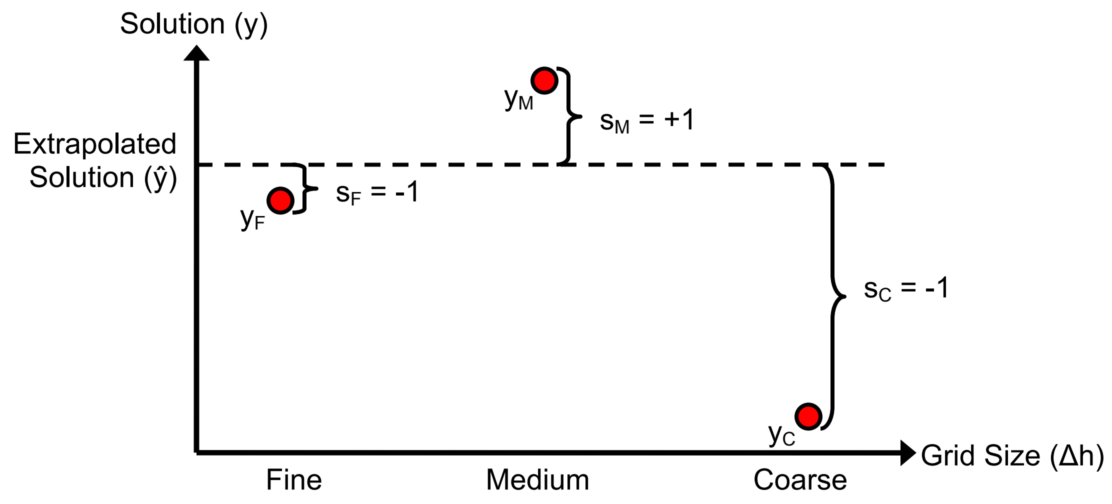
Solution for the quadratic equation can be written analytically as follows:

$$\begin{gathered} \Delta = (y_C - y_F)^2 - 4 s_C s_F (y_F - y_M)(y_M - y_C) \\ X^+ = \frac{-s_M(y_C - y_F) + \sqrt{\Delta}}{2 s_C (y_F - y_M)} \quad \text{and} \quad X^- = \frac{-s_M(y_C - y_F) + \sqrt{\Delta}}{2 s_C (y_F - y_M)} \end{gathered} \tag{4-43}$$

It is straightforward to calculate X^+, X^- and the corresponding orders of convergence for various combinations of signs in Table 4-1. Incoherent solutions, such as $\Delta < 0$, $X^+ < 0$, $X^- < 0$, indicate that the corresponding hypotheses are not supported by the data y_C, y_M, and y_F. If the assumption of non-monotonic error Ansatz model (4-20) is appropriate, then there should be a single combination of signs (s_C; s_M; s_F) that leads to a coherent solution for (p; β_1; $\hat{y}$).



(a) Data y_C, y_M, y_F and extrapolation $\hat{y}$ that correspond to hypothesis (H_4).



(b) Data y_C, y_M, y_F and extrapolation $\hat{y}$ that correspond to hypothesis (H_3).

Figure 4-3. Illustration of the ambiguity of non-monotonic solution convergence.

The reader is nevertheless cautioned that ambiguity may exist. This is shown in Figure 4-3 that illustrates three data points y_C, y_M, y_F and the corresponding extrapolation $\hat{y}$. The upper Figure 4-3(a) shows an acceptable solution for (p; β_1; $\hat{y}$) that corresponds to hypothesis (H_4): the absolute distance between numerical solutions and the extrapolation decreases with Δh and the signs are (s_C; s_M; s_F) = (-1: +1; +1). The lower Figure 4-3(b) shows another acceptable solution for (p; β_1; $\hat{y}$) that corresponds to hypothesis (H_3): the absolute distance also decreases with Δh while the signs are (s_C; s_M; s_F) = (-1: +1; -1). Should such situation occur, additional criteria would have to be relied upon to decide which solution needs to be selected.

The derivations presented in Section 4.6 extend the monotonic error Ansatz model applied to scalar predictions, $y_{h,t} = \hat{y} + \beta_1(\Delta h)^p$, to the case of non-monotonic convergence. Closed-form solutions are obtained in the special case of uniform grid refinement. The procedure applies to the 2-grid case, where only two grid resolutions are needed because the exact solution is known, and the N-grid case, $N \geq 3$. The only difficulty resides in the growth of the number of equations (4-40) that must be solved, that grows as $(2)^N$. To the best of the author's knowledge, no other generalization of the error Ansatz model $y_{h,t} = \hat{y} + \beta_1(\Delta h)^p$ is possible that would yield closed-form solutions.

The derivations presented in Section 4.6 extend the monotonic error Ansatz model applied to scalar predictions, $y_{h,t} = \hat{y} + \beta_1(\Delta h)^p$, to the case of non-monotonic convergence. Closed-form solutions are obtained in the special case of uniform grid refinement. The procedure applies to the 2-grid case, where only two grid resolutions are needed because the exact solution is known, and the N-grid case, $N \geq 3$. The only difficulty resides in the growth of the number of equations (4-40) that must be solved, that grows as $(2)^N$. To the best of the author's knowledge, no other generalization of the error Ansatz model $y_{h,t} = \hat{y} + \beta_1(\Delta h)^p$ is possible that would yield closed-form solutions.

4.7 Summary of Assumptions Made

As much as it is to introduce the neophyte to the main equations and procedures of code and solution verification, the purpose of this section is also to unravel the assumptions upon which it is based. The most important assumptions and misconceptions encountered are summarized below:

- *Section 4.1:* Verification activities can be categorized into code verification or solution verification. The main objective of the former is to provide assurance that the PDE are implemented, discretized, and solved correctly by comparing one or several numerical approximations to the exact solution of the continuous equations. Rigorous code verification therefore relies on the knowledge of the exact solution. The main objective of solution verification and self-convergence study is to verify that the rate of convergence observed by obtaining numerical solutions from several grid refinements matches the theoretical convergence rate of the numerical method.
- *Section 4.1:* The ultimate objective of solution verification is to estimate the numerical error of the discretized solution obtained from a given grid, relative to the exact solution of the continuous PDE. Pre-requisites to do so are that: 1) the code has been verified; 2) the actual order of convergence has been estimated; and 3) an appropriate error Ansatz model has been developed.
- *Section 4.1:* The most commonly encountered error anzatz model takes the form of an equation such as $y_{h,t} = \hat{y} + \beta_1(\Delta h)^p$ where $\hat{y}$ represents an estimate of the exact solution of the continuous equations. (Use $\hat{y} = y^*$ if the exact solution is known.) Although it is simple, intuitive, and allows for closed-form derivations, there is no rigorous foundation for this simplified error Ansatz model.
- *Section 4.1:* A general-purpose error Ansatz model is $e_h = \beta_1(\Delta h)^p$ where $e_h = ||\hat{y} - y_{h,t}||$ denotes a norm of the difference between the numerical solution ($y_{h,t}$) and the reference solution ($\hat{y}$). Choosing the absolute value, $e_h = |\hat{y} - y_{h,t}|$, is justified when the numerical method implemented discretizes a strong formulation of the equations, that is, the equations are enforced point-wise in the computational domain.
- *Section 4.1:* When a weak formulation of the PDE is discretized, the norm that defines the solution error is often the L^p norm. (Common choices are $p = 1, 2$, or ∞.) Because all norms defined over a finite dimensional space are equivalent, the choice of norm is not critical. It is argued, however, that the only choice consistent with the logic of searching

for a weak solution that is equivalent to the strong solution of the PDE is the so-called energy norm of the weak formulation.

- *Section 4.2:* The general-purpose error Ansatz model $e_h = \beta_1(\Delta h)^p$ and the simplified version $y_{h,t} = \hat{y} + \beta_1(\Delta h)^p$ both assume that the coupling between grid spacing Δh and time stepping Δt can be ignored, either because there is none or because Δt is kept constant during all simulations. In the latter case, Δt need to satisfy the most stringent CFL, stability, or accuracy condition, which usually means using a value determined by the characteristics of the finest mesh.
- *Sections 4.2 and 4.3:* It is necessary that the exact solution y* be known explicitly to obtain a closed-form solution for the unknowns (p; β_1) of the error Ansatz $e_h = \beta_1(\Delta h)^p$, no matter how the error e_h is defined. This applies whether the solutions are scalars or field variables (y^*, $y_{h,t} \in \Re^N$, for any $N \geq 1$). No closed-form solution exists when the numerical solutions are field variables and the corresponding exact solution is unknown.
- *Section 4.4:* A closed-form solution can be found for the triplet of unknowns (p; β_1; $\hat{y}$) of the simplified error Ansatz model $y_{h,t} = \hat{y} + \beta_1(\Delta h)^p$ where the exact solution is unknown and, therefore, replaced by an estimate $\hat{y}$. The error Ansatz model and its solution rely on three assumptions: 1) the solutions $\hat{y}$ and $y_{h,t}$ are scalar quantities; 2) convergence is strictly monotonic; and 3) the grid refinement ratio is constant. The third assumption can be relaxed at the cost of having to solve a non-linear optimization problem for the order of convergence. The second assumption can be relaxed and a semi-analytical solution procedure is derived in Section 4.6 to calculate ($\hat{y}$; p; β_1).
- *Section 4.5:* The grid convergence index is a standardized metric to report the results of solution verification and self-convergence studies. Small values of the GCI indicate a small error between the numerical and reference solutions. Constant values of the ratio $GCI/(\Delta h)^p$ obtained from several grids indicate asymptotic convergence. However the idea that the GCI defines a confidence interval of grid convergence uncertainty is simply wrong, no matter which value of the safety factor is used. The idea that the GCI defines 2-σ uncertainty bounds is also wrong.

5. Approach Proposed to Develop Non-linear Error Ansatz Models

Summary: Solution convergence is generally analyzed by postulating an error model that describes how the solution error behaves in space and, possibly, in time. Model parameters such as the regression coefficient β_1 and exponent p of an effect such as $\beta_1(\Delta h)^p$ are calibrated to improve the goodness-of-fit of the error Ansatz equation. Model calibration can be performed in a least-squares sense or using a non-linear optimization solver. The approach proposed here for model fitting combines the statistical concepts of effect screening and Bayesian model fitting. Prior probabilities are defined for each effect, such as $\beta_1(\Delta h)^p$ or $\beta_2(\Delta t)^q$, potentially included in the model. Models are formulated by randomly selecting effects according to the prior probabilities. Models visited are evaluated with a goodness-of-fit metric that assesses their ability to match the observed solution errors. The goodness-of-fit metric is treated as a likelihood function and, according to the Bayesian rule of probability, it is combined to prior probabilities to produce a posterior probability for each effect. The search algorithm implements a simple version of the Markov Chain Monte Carlo sampling that takes a random walk through all potential models. A consequence of Bayesian updating is that the more significant effects tend to be included more often which, in turn, improves the model goodness-of-fit. This procedure combines effect screening to model fitting, which is a step towards questioning the functional form of a model as well as fitting its parameters.

Section 5 describes the approach proposed to study the adequacy of formulating solution convergence error with a family of polynomial models parameterized by regression coefficients ($\beta_o;\beta_1;\beta_2;\beta_3$) and exponents (p;q;r;s):

$$e_h = \left\| y^* - y_{h,t} \right\|_{L^p} = \beta_o + \beta_1 (\Delta h)^p + \beta_2 (\Delta t)^q + \beta_3 (\Delta h)^r (\Delta t)^s + H.O.T. \quad (5\text{-}1)$$

At a high level, this problem is simply one of calibration (also known as "model fitting") where parameters of the model are optimized to make the predictions of equation (5-1) match observations with greater fidelity-to-data. The approach proposed, however, takes a small step towards calibrating, not only the model parameters ($\beta_o;\beta_1;\beta_2;\beta_3$;p;q;r;s), but also the functional form of the model itself. This is accomplished through the concept of statistical effect screening.

After briefly defining the main concepts in Section 5.1, the conventional approach to model fitting is overviewed for completeness in Section 5.2. Section 5.3 guides the reader though the main steps of Bayesian effect screening and model fitting [15]. Section 5.4 presents an application of the algorithm. Details about the procedure can be obtained from Reference [15].

5.1 Model Fitting and the Notion of Effect

Model fitting generally refers to the calibration of model parameters, labelled as β, given a sequence of data points $\{x_d;y_d\}$, d = 1 … N_{Data}, where N_{Data} denotes the number of observations. "Observation" here refers to an evaluation of the code that provides a numerical solution to be compared with the exact solution of the continuous equations. For the purpose of keeping the discussion simple, the model that builds an input-output relationship between inputs x and outputs y is treated as a black-box and denoted as:

$$y = M(x;\beta) \tag{5-2}$$

According to the "conventional" paradigm for model fitting, uncertainty about the knowledge of such input-output relationship is associated with model parameters β and the functional form of the model is generally not questioned. Note, however, that some of the model parameters can be defined in such a way that they modify the functional form of equation (5-2).

Another notion that must be clarified before proceeding is the notion of "effect". Broadly speaking, an effect is a term whose contribution to the model can be isolated and separated from the contribution of other effects. For example, a model that depends on three main effects, symbolically labelled "A", "B", and "C" for simplicity, is written as:

$$y = \beta_1 A + \beta_2 B + \beta_3 C \tag{5-3}$$

where the individual terms A, B, and C are either constants or functions of the inputs x. Main effects such as A and B may be combined into what is commonly called a linear interaction such as AB. Higher-order interactions would typically assume functional forms such as A^2B or A^2B^2.

Consider the two-input, one-output non-linear model:

$$y = 0.3 x_1 + 2.0 \sin(x_2) - 1.5 e^{-x_1 x_2} + 0.7 (x_1)^2 \sin(x_2) \tag{5-4}$$

This equation depends on two inputs $(x_1;x_2)$, and it is obtained as the combination of four separate effects. The effects are defined through three functions A, B, and C that depend on the pair $(x_1;x_2)$:

$$A = x_1, \quad B = \sin(x_2), \quad C = e^{-x_1 x_2} \tag{5-5}$$

Using the definitions (5-5), it can be verified easily that equation (5-4) is equivalent to:

$$y = 0.3 A + 2.0 B - 1.5 C + 0.7 A^2 B \tag{5-6}$$

The model is written above as the combination of three main effects, A, B, and C, and a non-linear interaction, A^2B. Effects are arbitrary functions that can assume any form (linear or non-linear, polynomial or not, etc.) of the inputs x. Note that, even though equation (5-6) *"looks like a polynomial function"*, it is nevertheless a non-linear function of the original inputs $(x_1;x_2)$.

In the remainder of Section 5, black-box models are written as a combination of effects:

$$y = \sum_{k=1 \ldots N_{Effects}} \beta_k X_k = X^T \beta \tag{5-7}$$

where X_k denotes a particular effect; $N_{Effects}$ is the total number of effects included in the model; and X and β are matrix-like notations for the effects and corresponding regression coefficients. Applied to equation (5-6), for example, the length-four vectors X and β are X = {A; B; C; A^2B} and β = {0.3; 2.0; -1.5; 0.7}. The polynomial-like appearance of equation (5-7) should not mask the fact that it actually applies to a wide range of models, and not just polynomials.

It is also emphasized that, while the input variables x may be independent, the effects X_k are neither independent nor uncorrelated. The Bayesian model screening discussed below does not require effects to be independent or uncorrelated.

What is most interesting about effects is to study their influence on predictions of the model. Omitting a significant effect would result in modifying significantly the prediction, while omitting a non-significant effect would not change the prediction by much. The connection between model fitting and effects is that, in order for the model to reproduce the available observations with fidelity, all significant effects must be included in the functional form of the model. Effect screening refers to the identification of the most significant effects given a design of experiments that prescribes the values of inputs and corresponding observations $\{x_d;y_d\}$ for d = 1 … N_{Data}.

5.2 Conventional Approach to Model Fitting

Equation (5-7) introduces the general form of a model that can be written as $y = X^T\beta$ after decomposition according to $N_{Effects}$ effects. According to this representation, the functions X_k are evaluated from the values of inputs x, and parameters β_k are constant. As previously mentioned, uncertainty about the model is assumed to be associated to uncertainty about the value of β.

Parameter calibration consists of estimating the value of parameters β such that predictions of the model (5-2) or, equivalently, (5-7) reproduce the available observations $\{x_d;y_d\}$ with fidelity. A commonly encountered practice to best-fit the parameters is to define an objective function, C, that represents the prediction error, and to minimize it using an optimization solver. One of the most straightforward choices of cost function is the Euclidean (L^2) norm of the prediction error:

$$C^2 = \sum_{d=1 \ldots N_{Data}} \left(y_d - X^{(d)T}\beta\right)^T \left(y_d - X^{(d)T}\beta\right) = e^T e \qquad (5\text{-}8)$$

where $X^{(d)}$ denotes the effect matrix evaluated at the dth observation or data point x_d; and e is a vector of length N_{Data} that stores the prediction error, that is, $e_d = (y_d - X^{(d)T}\beta)$. The Best, Linear, and Unbiased Estimator (BLUE) of parameters β is provided by the solution:

$$\hat{\beta} = (X^T X)^{-1} X^T y \qquad (5\text{-}9)$$

where the column vector y collects N_{Data} observations, and the N_{Data} rows-by-$N_{Effects}$ columns matrix X evaluates the $N_{Effects}$ effects for each of the N_{Data} observations:

$$y = \begin{Bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N_{Data}} \end{Bmatrix}, \quad X = \begin{bmatrix} X_1^{(1)} & X_2^{(1)} & \cdots & X_{N_{Effects}}^{(1)} \\ X_1^{(2)} & X_2^{(2)} & \cdots & X_{N_{Effects}}^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ X_1^{(N_{Data})} & X_2^{(N_{Data})} & \cdots & X_{N_{Effects}}^{(N_{Data})} \end{bmatrix} \qquad (5\text{-}10)$$

Clearly, other objective functions would yield different estimators. The generalization of the objective function (5-8) is commonly referred to as the Generalized Least-Squares (GLS) estimator [26]. Weighting matrices are introduced and a regularization term penalizes solutions too distant from the user-defined starting value $\beta^{(0)}$ of parameters β. Equations (5-11) and (5-12) show the GLS objective function and the corresponding GLS estimator, respectively:

$$C^2 = e^T W_{ee}^{-1} e + \lambda\left(\beta - \beta^{(0)}\right)^T W_{bb}^{-1}\left(\beta - \beta^{(0)}\right) \qquad (5\text{-}11)$$

$$\hat{\beta} = \left(X^T W_{ee}^{-1} X + \lambda W_{bb}^{-1}\right)^{-1} X^T W_{ee}^{-1} y \qquad (5\text{-}12)$$

where λ denotes a user-defined regularization parameter.

In general, the weighting matrices W_{ee} and W_{bb} are chosen arbitrarily or based on experience, for example, to weight the importance of some observations or parameters more than others. When covariance matrices are used, normal distributions are assumed to describe the uncertainty, and λ = 1, the GLS estimator of equation (5-12) can be interpreted as a Bayesian estimator. Rigorously speaking, other factors should appear in the definition of the Bayesian objective function. Because these additional factors are constant, however, the same estimator as the one shown in equation (5-12) is obtained. An important benefit of Bayesian inference is that it provides a posterior estimate of the covariance matrix:

$$\hat{W}_{bb}^{(posterior)} = \left(W_{bb}^{-1} + X^T W_{ee}^{-1} X\right)^{-1} \qquad (5\text{-}13)$$

Correlation coefficients of the posterior covariance matrix (5-13) provide insight into the quality of the estimator. One generally concludes that the functional form of the model is inappropriate when significant posterior correlation values are obtained between parameters that have no physical reason to be correlated [9]. With the exception of investigating the posterior correlation, however, no practical tool is available to select the appropriate form of a non-linear model, which is the process we refer to a model screening. Model form such as, for example, replacing a linear term by a cubic effect, is usually selected based on experience or empirical observation. Sometimes, several choices seem equally likely and the analyst has to go through the painstaking process of fitting each model and assessing their goodness-of-fit. Because it is based on the concept of goodness-of-fit, such approach can lead to over-fitting.

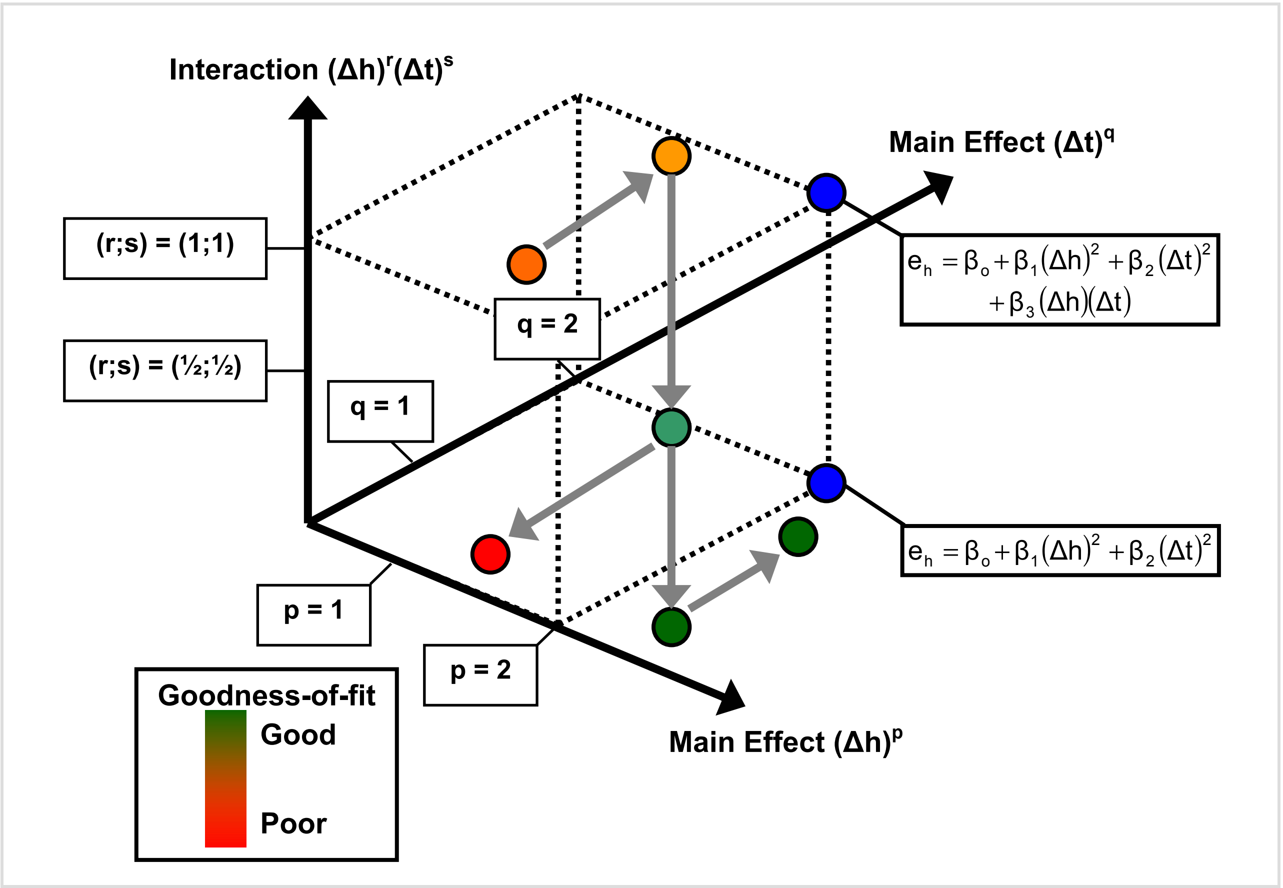
Another subtle but important issue is to estimate the posterior probability of a particular model as opposed to simply relying on the goodness-of-fit. By definition, the posterior probability is conditioned on the evidence available, that is, the observations or data points. Posterior probability and goodness-of-fit complement each other because the former indicates if the analyst's prior opinion of the form of the model is consistent with the evidence. In Section 5.3, a tool is proposed for model screening based on the concept of posterior probability.

5.3 Bayesian Effect Screening and Model Fitting

In the previous section, the state-of-the-practice in model fitting is briefly overviewed. A polynomial-like decomposition in effects is considered for simplicity. It is emphasized that this formalism does not imply that the Bayesian model screening technique summarized here is restricted to polynomials. Each effect of the decomposition can be an arbitrarily complex and non-linear function of the input variables x. Essentially, the only two assumptions made are as follows. First, a model $y = M(x;\beta)$ must be defined. Second, an inference algorithm is available for calibrating the parameters β. The inference is usually referred to as "best-fitting" when applied to polynomials, or "training" when applied to neural networks. In this study, the calibration of model parameters is handled through a least-squares solver when the model can be decomposed according to equation (5-7) or a non-linear optimization solver when it cannot.

Model screening consists in identifying the most probable models based on a family of models defined by the user and given observations that the model must reproduce with the highest possible degree of fidelity. It is emphasized that model screening does not necessarily identify the best model but rather ranks all potential contributing effects according to their posterior probability of occurrence.

The procedure starts by, first, defining a family of models. This is achieved by defining various effects X_k and how these effects are allowed to interact to form the population of potential models. Figure 5-1 illustrates the concept of a family of models by showing the three effects, $X_1 = (\Delta h)^p$, $X_2 = (\Delta t)^q$, and $X_3 = (\Delta h)^r(\Delta t)^s$, in the case of solution verification where the model is a function of space and time discretization variables Δh and Δt. The model-forming rule illustrated in Figure 5-1 is that the three effects are allowed to be combined in proportions defined by the regression parameters β_1, β_2, and β_3. Each model visited is evaluated by calculating its goodness-of-fit to the observations. Equations (5-8) and (5-11) are examples of fidelity metrics used in Sections 6 and 7. This is illustrated in Figure 5-1 with dots whose colors indicate the degree of fidelity-to-data. The likelihood that a particular model is appropriate to represent the data is indicated by its value of the goodness-of-fit metric. It is this notion of likelihood that is employed to guide the search for the most appropriate model(s) and, hence, the identification of the most significant effects to be included in the model(s).



Interaction $(\Delta h)^r(\Delta t)^s$
Main Effect $(\Delta t)^q$
(r;s) = (1;1)
(r;s) = (½;½)
q = 2
q = 1
$e_h = \beta_o + \beta_1(\Delta h)^2 + \beta_2(\Delta t)^2 + \beta_3(\Delta h)(\Delta t)$
$e_h = \beta_o + \beta_1(\Delta h)^2 + \beta_2(\Delta t)^2$
p = 1
p = 2
Main Effect $(\Delta h)^p$
Goodness-of-fit
Good
Poor

Figure 5-1. Definition of a family of models with effects $(\Delta h)^p$, $(\Delta t)^q$, and $(\Delta h)^r(\Delta t)^s$.

The second step of the procedure is to assign the prior probability of occurrence for each effect X_k. The priors can reflect empirical observations, experience or the analyst's knowledge of the phenomenon investigated. In the application discussed in Sections 6 and 7, for example, no specific knowledge can be used to guide a pertinent choice of priors. Probabilities of occurrence are therefore set to a uniform 25% level for all main effects.

The third step is to let the Bayesian screening method find the most appropriate models by searching through all possible combinations of effects. The most probable models are those that best reproduce the data. Fidelity-to-data can be assessed using the Root Mean Square (RMS) error between observations and predictions, which is equivalent to equation (5-8). Assuming Gaussian distributions, the RMS error becomes proportional to the likelihood function L(y|β) that estimates the appropriateness of the model given the knowledge of parameters β. Other functions can be used to define the likelihood, such as the Bayesian objective function (5-11) and many other functions commonly used in test-analysis correlation and model calibration [18].

Once the likelihood of a particular model has been estimated, the posterior probabilities of its effects can be updated according to the Bayes Theorem that states that PDF(β|y), the posterior Probability Density Function (PDF), is equal to the likelihood function L(y|β) multiplied by the prior probability PDF(β) and divided by the probability PDF(y) of the data:

$$\mathrm{PDF}(\beta \mid y) = \frac{L(y \mid \beta)\mathrm{PDF}(\beta)}{\mathrm{PDF}(y)} \qquad (5\text{-}14)$$

The probability PDF(y) of observations is generally kept constant and omitted in equation (5-14). A simple re-normalization suffices to correctly scale the probability distribution. Because the procedure is iterative in nature, the Bayes update (5-14) is repeated and posteriors of the nth iteration become the priors of the (n+1)th iteration. All models visited are kept in memory and, once enough samples have been drawn, the probability of occurrence of each model is estimated by its frequency of occurrence, that is, the ratio between the number of times the model is visited and the total number of iterations. The procedure is summarized in Figure 5-2.

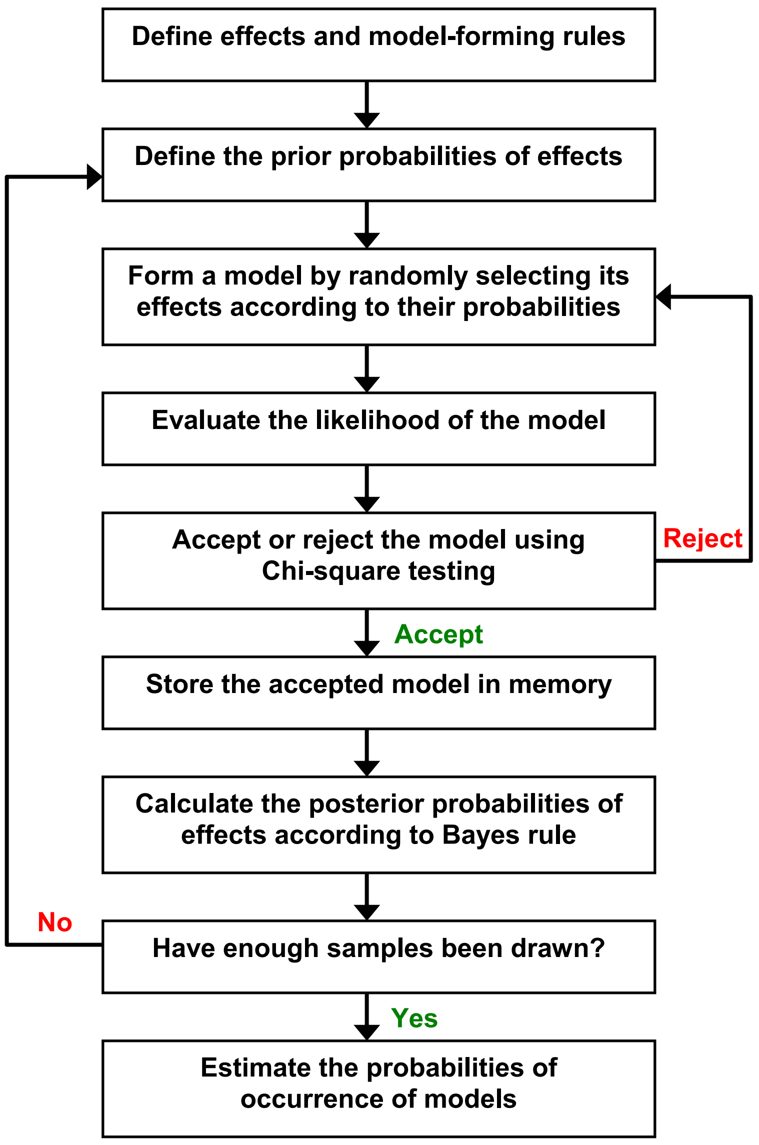

Define effects and model-forming rules
Define the prior probabilities of effects
Form a model by randomly selecting its effects according to their probabilities
Evaluate the likelihood of the model
Accept or reject the model using Chi-square testing
Reject
Accept
Store the accepted model in memory
Calculate the posterior probabilities of effects according to Bayes rule
No
Have enough samples been drawn?
Yes
Estimate the probabilities of occurrence of models

Figure 5-2. Flow diagram of the Bayesian screening and model fitting algorithm.

The output of the Bayesian model screening algorithm illustrated in Figure 5-2 is, first, the probabilities of occurrence of the most appropriate members of the user-defined family of models; second, the goodness-of-fit indicators for each model selected; and, third, the posterior probabilities PDF(β|y) of effects involved in the most likely models.

To achieve this result, however, the unknown likelihood function L(y|β) must be sampled. The problem of exploring an unknown PDF is solved with the Markov Chain Monte Carlo (MCMC) algorithm. The MCMC sampling is advantageous in this situation because it can sample any distribution, whether it is Gaussian or not. The MCMC sampling can be viewed conceptually as an optimization solver that performs a random walk through the optimization space. This concept is illustrated in Figure 5-1 where the candidate models, represented symbolically as points of different colors in the optimization space, are visited sequentially. More appropriate models are guaranteed more frequent visits because the acceptance criterion of a given model is based on its likelihood function. A model is accepted if its value of the likelihood function passes a Chi-square test. This acceptance criterion implies that inappropriate models have a small chance of being accepted just like appropriate models have a small chance of rejection. If rejected, a new model is selected randomly in the neighborhood of the last accepted model. The sequence of models accepted is stored in memory to estimate, once the process has been completed, the probability of occurrence of each model.

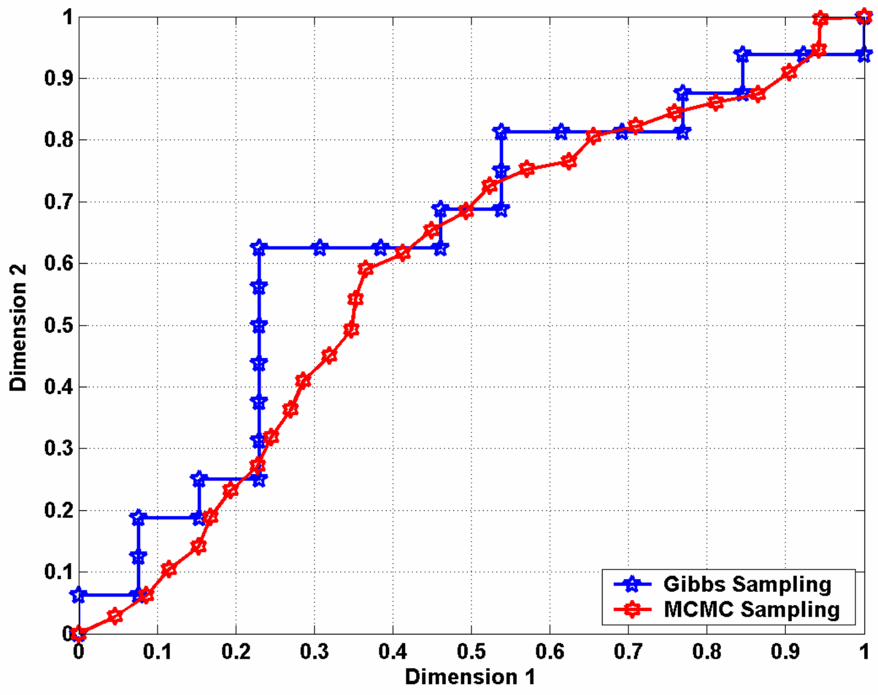


Figure 5-3. An example of random walks obtained with the Gibbs and MCMC samplers.

The sampling procedure used in this study is the Gibbs sampler, the simplest of the many variants of the MCMC algorithm. The main difference between the two is that the Gibbs algorithm samples one direction of the search space at a time, which makes for simpler numerical implementation. Figure 5-3 illustrates the difference between MCMC and Gibbs sampling. It pictures two random walks from the lower left corner (X=0; Y=0) to the upper right corner (X=1; Y=1). A constraint is enforced that prevents the 30 points drawn in both sequences from being repeated and from moving backwards. Blue pentagram symbols show a sequence of Gibbs samples while red hexagram symbols picture a realization of the MCMC chain. In the former case, the solution is advanced in one direction at a time whereas the MCMC chain randomly advances the solution in the two dimensions simultaneously.

5.4 Numerical Example of Model Screening

A simple application is presented to illustrate the overall performance of the model screening procedure. Consider an output variable y defined by the following input-output model:

$$y = 2\sin(2t) + 3\cos(t) - 1.5\sin(3t)\cos(2t) \tag{5-15}$$

where t is an input variable that varies from zero to 50 sec. with increments of $\Delta t = 0.05$ sec. It is assumed that the functional form of the model shown in equation (5-15) is unknown. Instead, sample data $y_d = y(d.\Delta t)$, for d = 0 … 100 points, are collected. The problem is to identify the numerical model that best matches the observations. The continuous solution (5-15) is shown in Figure 5-4 with a blue solid line and red hexagram symbols represent the data samples.

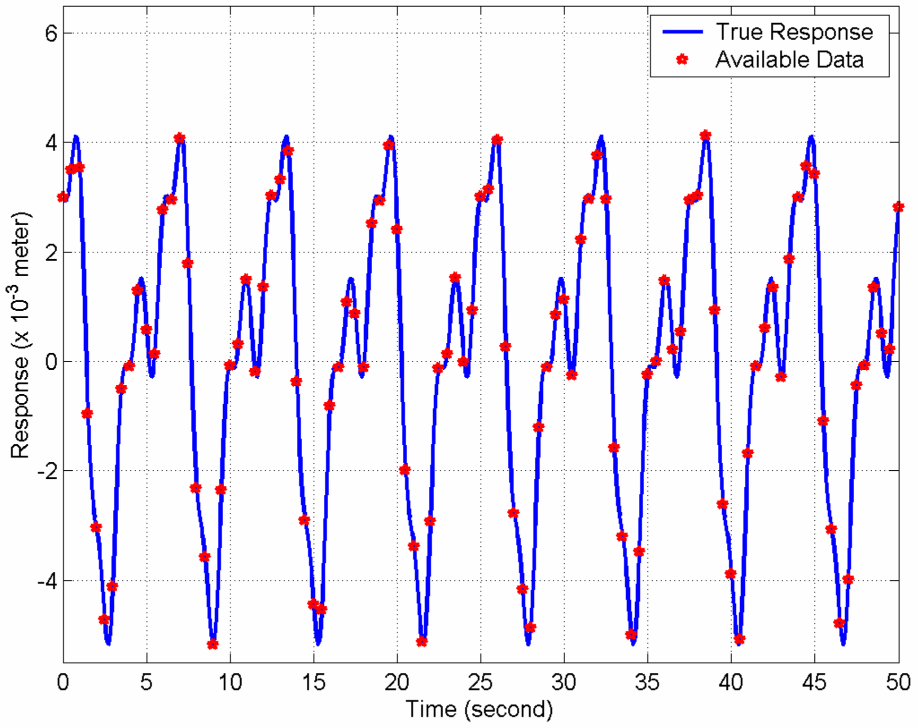


Figure 5-4. Data samples used to estimate an unknown functional form.

Next, consider a set of candidate predictors (or main effects):

$$\begin{cases} X_1 = \sin(t) \\ X_2 = \cos(t) \\ X_3 = \sin(2t) \\ X_4 = \cos(2t) \\ X_5 = \sin(3t) \\ X_6 = \cos(3t) \end{cases} \tag{5-16}$$

In addition to the six predictors of equation (5-16), six other predictors labelled X_7, X_8, X_9, X_{10}, X_{11} and X_{12} are defined as random functions. It can be verified easily that, if the functional form of the output variable y were known, then it could be written exactly as:

$$y = 3X_2 + 2X_3 - 1.5X_4X_5 \tag{5-15}$$

Clearly, y does not depend on predictors X_1, X_6, X_7, X_8, X_9, X_{10}, X_{11} and X_{12}. The objective of model screening is to identify the model form (5-15). Equivalently, it can be stated that the objective of model screening is to identify the linear effects X_2, X_3 and the linear interaction effect X_4X_5 from all the potential combinations defined by the family of models considered.

The family of models defined for this illustration is composed of the linear models that include the twelve linear effects X_k and the linear interaction models, defined as the previous models augmented with the 66 interaction effects X_pX_q for $p \neq q$. The total number of different effects X_k and X_pX_q with twelve predictors is equal to 78. The total number of different models that can be defined belonging to this family by combining the 78 effects is more than $3.02.10^{+23}$ models, a rather large number to explore.

The procedure described in the foregoing section is applied to the data using 50 samples dedicated to the initialisation of the Gibbs sampler and 100 samples for the computation. Initializing the Markov chain is referred to as "burn-in" and guarantees that the remainder of the chain is not biased due to a particular choice of starting point. The samples drawn during burn-in are disregarded and only the 100 samples drawn during the optimisation itself are kept to estimate the final probability of occurrence of each model in the family. The top five models are listed in Table 5-1. It can be observed that the best model in terms of posterior model probability is the actual model. The mean square error for the top five models is about 0.003%. This means that it is not necessary to include other terms than the ones present in the best model.

Table 5-1. Top five models selected and their number of appearances.

Top Model	Effects	Posterior Probability
1	X_2,X_3,X_4X_5	52.0%
2	X_2,X_3,X_4X_5,X_{10}	3.0%
3	X_2,X_3,X_4X_5,X_3X_4	2.0%
4	X_2,X_3,X_4X_5,X_2X_{10}	2.0%
5	X_2,X_3,X_4X_5,X_2X_5	2.0%

Figure 5-5 represents the marginal posterior probability of each effect being in a particular model. The prior probabilities, that reflect the prior knowledge, are set to 25% for main effects X_k; 10% for interaction effects X_pX_q if one of the parent effect X_p or X_q is selected in the model; and 1% only for interaction effects X_pX_q when neither X_p nor X_q are considered in the model. These uniform probabilities reflect the fact that little is known about the form of the model before starting the analysis. It can be observed that effects 2, 3 and 43, namely X_2, X_3 and X_4X_5, are associated with a probability of 100% while the other effects may be ignored because their posterior probabilities are reduced to insignificant levels.

In conclusion, the Bayesian model screening algorithm clearly suggests a model that includes the three effects X_2, X_3 and X_4X_5. The estimated regression coefficients corresponding to these effects are equal to 2.99, 2.02, and -1.52, respectively. They are in excellent agreement with the actual coefficients shown in equation (5-15). The algorithm implementation is a toolbox of interpreted MATLAB™ functions. Although not written to be performance-optimal, it performs the analysis in a few seconds of CPU time with a typical desktop personal computer.

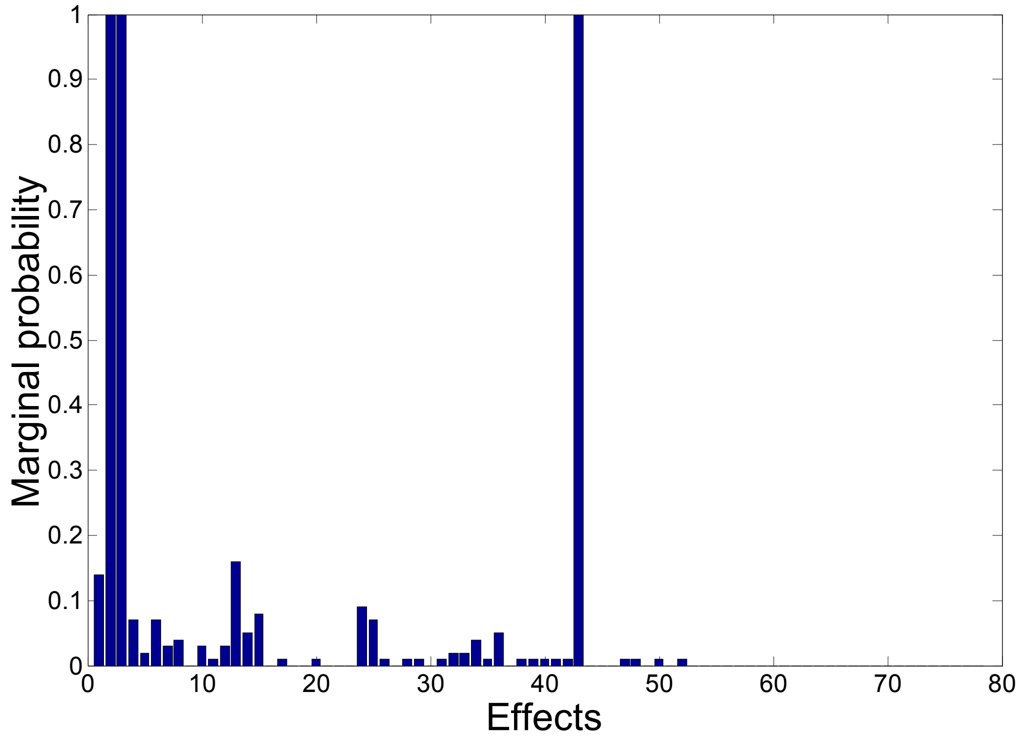


Figure 5-5. Posterior probabilities of main effects (1-12) and linear interactions (13-78).

6. Illustration With the Vortex Evolution Test Problem

Summary: The purpose of Section 6 is to illustrate the application of statistical techniques to develop error Ansatz models for solution convergence. The Euler equations of compressible gas dynamics are given because they govern the four code verification test problems used in Section 7. Results discussed in this section are obtained with the vortex evolution problem, a periodic flow that yields a non-linear but smooth solution. Solution convergence errors are given for the peak and root-mean-square values of the density, momentum, and total energy state variables. L^1, L^2, and L^∞ norms of the error are calculated over the entire computational domain. These data sets are used to develop, first, linear and space-only error Ansatz equations. Quadratic convergence in space is observed. Then, non-linear error Ansatz models are obtained using the statistical screening and fitting method described in Section 5. The use of posterior probabilities for effect screening is illustrated. The convergence rates observed are $1 \le p \le 2$ for the effect $(\Delta h)^p$ of spatial discretization, $q = -1$ for the effect $(\Delta t)^q$ of temporal discretization, and $(r;s) = (2;-1)$ for the non-linear coupling effect $(\Delta h)^r(\Delta t)^s$.

The statistical model fitting approach presented in Section 5 is illustrated with results from the vortex evolution code verification test problem. The vortex evolution problem, denoted by Vortex2D, features a periodic flow condition and smooth solution that nevertheless exercises the non-linearity of Euler equations of gas dynamics. The purpose of Section 6 is not so much to investigate the formulation of an error Ansatz model for solution convergence, but rather to illustrate the statistical tools applied here to model fitting. Section 6 also serves the purpose of introducing equations and notations so that they do not need to be repeated in Section 7.

Section 6.1 introduces the laws of conservation of gas dynamics that are common to the four code verification test problems discussed in Section 7. Section 6.2 defines the vortex evolution test problem and reports computational results. The development of space-only error Ansatz models for solution verification is reported in Section 6-3 and non-linear, space-time models are discussed in Section 6-4.

6.1 The Euler Equations of Gas Dynamics

A general description of Euler equations of compressible, inviscid, and non-heat conducting gas dynamics in one dimension (1D) or two dimensions (2D) is given in equation (6-1):

$$\frac{\partial q}{\partial t} + \frac{\partial F_x(q)}{\partial x} + \frac{\partial F_y(q)}{\partial y} = 0$$

$$q = \begin{Bmatrix} \rho \\ \rho U_x \\ \rho U_y \\ E \end{Bmatrix}, \quad F_x(q) = \begin{Bmatrix} \rho U_x \\ \rho U_x^2 + p \\ \rho U_x U_y \\ U_x(E+p) \end{Bmatrix}, \quad F_y(q) = \begin{Bmatrix} \rho U_y \\ \rho U_x U_y \\ \rho U_y^2 + p \\ U_y(E+p) \end{Bmatrix} \qquad (6\text{-}1)$$

where the five state variable are density (ρ); flow velocity (Ux;Uy); thermodynamic pressure (p); and total energy (E). Total energy E and Specific Internal Energy (SIE) e are related by:

$$E = \rho\left[e + \frac{1}{2}\left(U_x^2 + U_y^2\right)\right] \tag{6-2}$$

Since these four coupled equations in 2D feature a total of five unknowns (ρ; Ux; Uy; p; e), a closure equation is needed that relates pressure (or, equivalently, temperature) to density and internal energy. The polytropic gas Equation Of State (EOS) is assumed:

$$p = (\gamma - 1)\rho e \tag{6-3}$$

where γ is the adiabatic exponent of ideal gas. The temperature (T) and entropy (S) are defined as T = p/ρ and S = p/(ργ), respectively.

Equations (6-1) to (6-3) describe the conservation of mass, momentum, and total energy in an Eulerian frame for the dynamics of a single, compressible, inviscid, and non-heat-conducting gas. Although relatively simple, these equations can represent convergent, divergent, or periodic flow conditions depending on the choice of boundary and initial conditions. Because of their non-linearity and hyperbolic nature, equations (6-1) can result in smooth or discontinuous (shocked) solutions. They can also degenerate into the parabolic equation of wave propagation and yield a linear, smooth solution.

The four test problems analyzed in Section 7 are all described by the system of equations (6-1) to (6-3), even though they represent different flow conditions (convergent vs. divergent vs. periodic) and provide solutions of different natures (linear vs. non-linear, smooth vs. shocked). Results presented in Section 6 are obtained by analyzing the Vortex2D problem using an Adaptive Mesh Refinement (AMR) Eulerian hydro-dynamics code developed by Dr. Shengtai Li of the Los Alamos National Laboratory, code known as AMR-MHD. Results shown in Section 7 for the Noh (Noh1D and Noh2D), Sedov (Sedov1D and Sedov2D), vortex evolution (Vortex2D), and wave propagation (Wave2D) test problems are obtained with a Eulerian hydro-code developed under the Los Alamos Code Project "Crestone".

6.2 The Vortex Evolution Test Problem

The vortex evolution problem solves in two dimensions the Euler equations of compressible gas dynamics where the mean flow initialized at density ρ_o = 1 gm-cm^{-3}, pressure p_o = 1 dyne-cm^{-2}, and velocity $U_x = U_y$ = 1 cm-sec^{-1} is perturbed by an isentropic vortex [17]. The perturbation is defined in terms of velocity δU_x, δU_y, and temperature δT. It is parameterized by a strength parameter ε whose value is selected to be ε = 5:

$$\delta U_x = \frac{-\varepsilon y}{2\pi} e^{\frac{1}{2}\left(1-R^2\right)}, \quad \delta U_y = \frac{\varepsilon x}{2\pi} e^{\frac{1}{2}\left(1-R^2\right)}$$
$$\delta T = \frac{-(\gamma-1)\varepsilon^2}{8\gamma\pi^2} e^{-\left(1-R^2\right)} \tag{6-4}$$

where R denotes the radius, $R^2 = x^2 + y^2$. The solution is continuous and the perturbation introduced to the mean flow (ε = 5) is strong enough that it excites the non-linearity of Euler equations. The perturbation is not strong enough to develop a shock. Details of the problem setting and derivation of an analytical solution for the continuous equations (6-1) to (6-4) are available from Reference [17].

Results presented in Section 6 are obtained with calculations performed on three uniform grids with a coarse resolution of 80-by-80 = 6,400 cells or Δh_C = 12.50.10^{-2} cm; a medium resolution of 120-by-120 = 14,400 cells or Δh_M = 8.3$\underline{3}$.10^{-2} cm; and a fine resolution of 160-by-160 = 25,600 cells or Δh_F = 6.25.10^{-2} cm. Note that these grids do not provide a constant spatial refinement ratio since it is equal to R_H = 1.50 going from the coarse grid to the medium grid and it is equal to $R_H = 1.3\underline{3}$ going from the medium grid to the fine grid. CFL-limited time stepping is implemented with three Courant-Friedrichs-Levy (CFL) conditions, $N_{CFL} = 0.90$, $N_{CFL} = 0.45$, and $N_{CFL} = 0.04$. The CFL number, denoted by N_{CFL}, is approximately equal to:

$$N_{CFL} \approx \left(\frac{\Delta t}{\Delta h}\right)\max(\omega_{Sound}) \tag{6-5}$$

where $\max(\omega_{Sound})$ denotes the maximum speed of sound that the computational grid must be able to capture.

Because the flow is compressible, the maximum sound speed changes and N_{CFL} and the corresponding time steps Δt are not kept perfectly constant during the entire simulation. It is verified, however, that the variation is small (less than 10%) and only a few cycles need to use CFL numbers that vary significantly from the nominal settings of $N_{CFL} = 0.90$, $N_{CFL} = 0.45$, and $N_{CFL} = 0.04$. These target CFL conditions are therefore treated as constant in this analysis.

Other settings of the vortex evolution problem are that the size of the computational domain is [0; 10]-by-[0; 10] cm^2 and equations are integrated in time and space for a total duration of 100 sec. The density (ρ), momentum in the X-direction (ρU_x), momentum in the Y-direction (ρU_y), and total energy (E) state variables are outputted at the final time of 100 sec. The results presented below are for these four output fields.

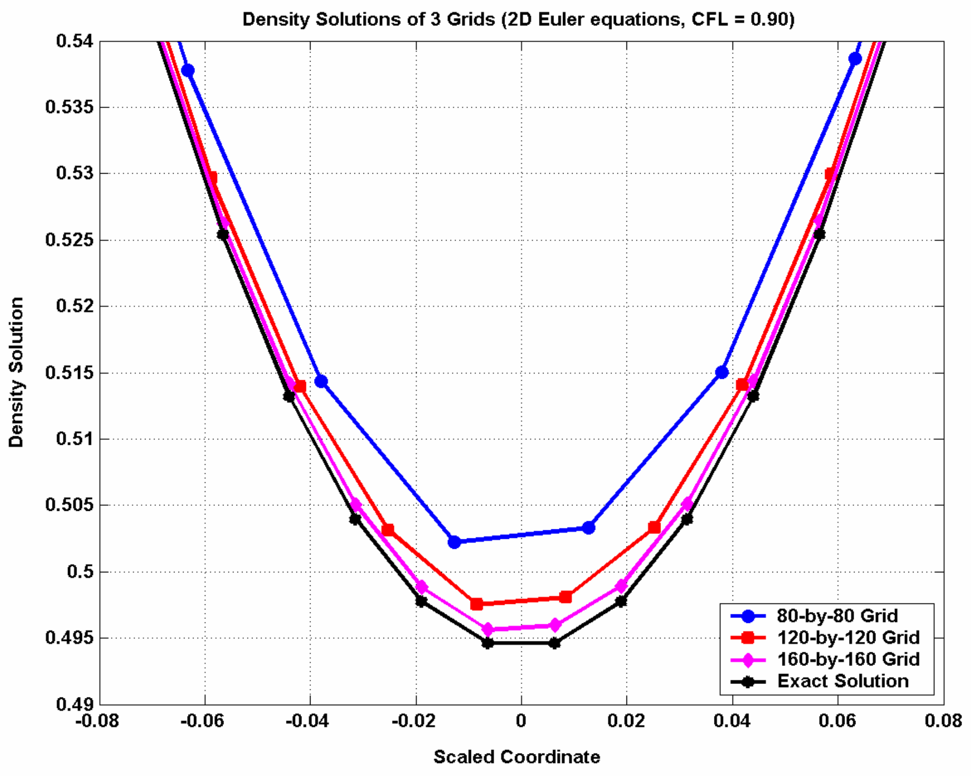


Figure 6-1. Analytical and numerical density fields of the Vortex2D problem.

Figure 6-1 compares three numerical solutions obtained for the density state variable to the exact solution over a small region of the computational domain, $4.6 \le X \le 5.4$ cm and Y = constant = 5 cm. The numerical solutions are obtained with $N_{CFL} = 0.90$. Comparison with the exact solution indicates convergence as $\Delta h \rightarrow 0$.

A typical issue of solution verification is also illustrated by Figure 6-1. The process of grid refinement produces successive cells that do not, except by chance, overlap with each other. A consequence is that the (unknown) solution is evaluated at different locations within each computational grid. Such offsets between points of the coarse, medium, and fine discretizations are clearly visible in Figure 6-1. One solution to handle this mismatch would be to spatially average all results from the fine and medium grids onto the coarse grid. This avenue is not followed here because it would result in somewhat of a loss of information.

Another concern, mentioned for completeness, is to output values estimated at the center of a computational cell. This commonly encountered practice is questionable for codes that rely on finite volume approximations [24]. The argument is summarized in the paragraph below:

> *"Technically, the exact solution at the center of the computational cell should not be used for codes that employ finite volume schemes, and not finite difference schemes, as does the RAGE code used in this study. Instead, the integrated average of the exact solution over the entire cell should be used. However, previous experience and numerous calculations using the RAGE code have shown little difference in the results between convergence calculations that use the exact solution at the zone center and calculations using the integral of the exact solution across the zone. We decided to use the value at the zone center for this study."* (From Reference [24].)

The same strategy is adopted in this study because the hydro-code used is designed to reach second-order accuracy. (Note, however, that using averages vs. point-wise values of the exact solution could be a potential issue if the numerical solver were third-order accurate.)

Table 6-1 compares the exact peak values of state variables (ρ; ρU_x; ρU_y; E) to the peak values obtained from numerical calculations. Table 6-2 provides a similar comparison for the Root Mean Square (RMS) values. In both cases, the top section of tables lists values extracted from the solutions of the continuous equations; the middle section lists values extracted from the numerical solutions; and the bottom section lists the differences between exact and numerical values. Differences δ are expressed as percentages of the exact values:

$$\delta = 100\left(\frac{z^* - z_{h,t}}{z^*}\right)\% \qquad (6\text{-}6)$$

where "z" denotes a feature of the exact or numerical solution, here, either the peak or RMS value. A peak value is a point-wise quantity, that is, a value obtained at a single cell of the computational domain, while a RMS value provides an average over the entire domain. To make averages independent from the number of cells upon which they are calculated, RMS values are scaled to the number of cells defined in a particular grid:

$$\mathrm{RMS} = \sqrt{\frac{1}{N_{Cells}} \sum_{k=1\cdots N_{Cells}} y^2(k)} \qquad (6\text{-}7)$$

where N_{Cells} is the number of cells; "y" denotes either the exact solution ($y = y^*$) or numerical solution ($y = y_{h,t}$); and "k" is the cell identifier, k = 1 … N_{Cells}.

It is observed from Tables 6-1 and 6-2 that agreement between the exact and numerical solutions is excellent, up to the second digit for all state variables and calculations. Relative errors are less than 1% in all cases, with only three peak value errors greater than 0.5% and only three RMS value errors greater than 0.1%. Convergence to the exact solution as the grid is refined, $\Delta h \rightarrow 0$, is clearly visible for the peak and RMS values. It can also be observed that refining the time step, $\Delta t \rightarrow 0$, which corresponds to running the hydro-calculation at lower CFL numbers, tends to increase the solution error.

Table 6-1. Exact and numerical solutions for peak values (Vortex2D).

Run Settings		**Exact Solutions for Peak Values**			
Cell Size, Δh (cm)	**N_{CFL} (Unitless)**	**Density, ρ (gm-cm^{-3})**	**Momentum, ρU_x (gm-cm^{-2}-sec^{-1})**	**Momentum, ρU_y (gm-cm^{-2}-sec^{-1})**	**Energy, E (erg-gm^{-1})**
$12.50.10^{-2}$	0.90	1.0000	1.5437	1.5437	4.2542
$8.3\underline{3}.10^{-2}$	0.90	1.0000	1.5437	1.5437	4.2542
$6.25.10^{-2}$	0.90	1.0000	1.5437	1.5437	4.2542
$12.50.10^{-2}$	0.45	1.0000	1.5472	1.5472	4.2528
$8.3\underline{3}.10^{-2}$	0.45	1.0000	1.5472	1.5472	4.2528
$6.25.10^{-2}$	0.45	1.0000	1.5472	1.5472	4.2528
$12.50.10^{-2}$	0.04	1.0000	1.5465	1.5465	4.2539
$8.3\underline{3}.10^{-2}$	0.04	1.0000	1.5465	1.5465	4.2539
Run Settings		**Numerical Solutions for Peak Values**			
Cell Size, Δh (cm)	**N_{CFL} (Unitless)**	**Density, ρ (gm-cm^{-3})**	**Momentum, ρU_x (gm-cm^{-2}-sec^{-1})**	**Momentum, ρU_y (gm-cm^{-2}-sec^{-1})**	**Energy, E (erg-gm^{-1})**
$12.50.10^{-2}$	0.90	1.0002	1.5378	1.5380	4.2522
$8.3\underline{3}.10^{-2}$	0.90	1.0002	1.5336	1.5368	4.2499
$6.25.10^{-2}$	0.90	1.0002	1.5297	1.5340	4.2424
$12.50.10^{-2}$	0.45	1.0001	1.5460	1.5462	4.2529
$8.3\underline{3}.10^{-2}$	0.45	1.0001	1.5443	1.5461	4.2522
$6.25.10^{-2}$	0.45	1.0002	1.5416	1.5453	4.2487
$12.50.10^{-2}$	0.04	1.0001	1.5465	1.5466	4.2542
$8.3\underline{3}.10^{-2}$	0.04	1.0001	1.5456	1.5466	4.2527
Run Settings		**Relative Differences, δ = (Exact–Numerical)/Exact**			
Cell Size, Δh (cm)	**N_{CFL} (Unitless)**	**δ-error, ρ (x10^{-2}%)**	**δ-error, ρU_x (x10^{-2}%)**	**δ-error, ρU_y (x10^{-2}%)**	**δ-error, E (x10^{-2}%)**
$12.50.10^{-2}$	0.90	-2.00	38.22	36.92	4.70
$8.3\underline{3}.10^{-2}$	0.90	-2.00	65.43	44.70	10.11
$6.25.10^{-2}$	0.90	-2.00	90.69	62.84	27.74
$12.50.10^{-2}$	0.45	-1.00	7.76	6.46	-0.24
$8.3\underline{3}.10^{-2}$	0.45	-1.00	18.74	7.11	1.41
$6.25.10^{-2}$	0.45	-2.00	36.19	12.28	9.64
$12.50.10^{-2}$	0.04	-1.00	0	-0.65	-0.71
$8.3\underline{3}.10^{-2}$	0.04	-1.00	5.82	-0.65	2.82

A difference in convergence behavior of peak and RMS values is noticeable. Convergence of RMS values to the exact solution as $\Delta h \rightarrow 0$ in Table 6-2 seems “faster” than convergence of peak values in Table 6-1. This observation is due to the fact that convergence of a point-wise quantity, that is, a quantity estimated at a single computational cell, is more difficult to achieve than convergence of an average. This remark strengthens the claim that caution should be exercising when formulating point-wise error models such as those of equations (4-9) or (4-20).

Table 6-2. Exact and numerical solutions for RMS values (Vortex2D).

Run Settings		**Exact Solutions for RMS Values**			
Cell Size, Δh (cm)	**N_{CFL} (Unitless)**	**Density, ρ ($gm\text{-}cm^{-3}$)**	**Momentum, ρU_x ($gm\text{-}cm^{-2}\text{-}sec^{-1}$)**	**Momentum, ρU_y ($gm\text{-}cm^{-2}\text{-}sec^{-1}$)**	**Energy, E ($erg\text{-}gm^{-1}$)**
$12.50.10^{-2}$	0.90	0.9846	0.9949	0.9949	3.4620
$8.3\underline{3}.10^{-2}$	0.90	0.9846	0.9949	0.9949	3.4620
$6.25.10^{-2}$	0.90	0.9846	0.9949	0.9949	3.4620
$12.50.10^{-2}$	0.45	0.9846	0.9949	0.9949	3.4620
$8.3\underline{3}.10^{-2}$	0.45	0.9846	0.9949	0.9949	3.4620
$6.25.10^{-2}$	0.45	0.9846	0.9949	0.9949	3.4620
$12.50.10^{-2}$	0.04	0.9846	0.9949	0.9949	3.4620
$8.3\underline{3}.10^{-2}$	0.04	0.9846	0.9949	0.9949	3.4620
Run Settings		**Numerical Solutions for RMS Values**			
Cell Size, Δh (cm)	**N_{CFL} (Unitless)**	**Density, ρ ($gm\text{-}cm^{-3}$)**	**Momentum, ρU_x ($gm\text{-}cm^{-2}\text{-}sec^{-1}$)**	**Momentum, ρU_y ($gm\text{-}cm^{-2}\text{-}sec^{-1}$)**	**Energy, E ($erg\text{-}gm^{-1}$)**
$12.50.10^{-2}$	0.90	0.9844	0.9942	0.9942	3.4606
$8.3\underline{3}.10^{-2}$	0.90	0.9843	0.9940	0.9939	3.4601
$6.25.10^{-2}$	0.90	0.9842	0.9936	0.9936	3.4595
$12.50.10^{-2}$	0.45	0.9845	0.9947	0.9947	3.4616
$8.3\underline{3}.10^{-2}$	0.45	0.9845	0.9946	0.9946	3.4614
$6.25.10^{-2}$	0.45	0.9845	0.9945	0.9945	3.4612
$12.50.10^{-2}$	0.04	0.9846	0.9948	0.9948	3.4618
$8.3\underline{3}.10^{-2}$	0.04	0.9846	0.9948	0.9947	3.4618
Run Settings		**Relative Differences, δ = (Exact–Numerical)/Exact**			
Cell Size, Δh (cm)	**N_{CFL} (Unitless)**	**δ-error, ρ ($x10^{-2}\%$)**	**δ-error, ρU_x ($x10^{-2}\%$)**	**δ-error, ρU_y ($x10^{-2}\%$)**	**δ-error, E ($x10^{-2}\%$)**
$12.50.10^{-2}$	0.90	2.03	7.04	7.04	4.04
$8.3\underline{3}.10^{-2}$	0.90	3.05	9.05	10.05	5.49
$6.25.10^{-2}$	0.90	4.06	13.07	13.07	7.22
$12.50.10^{-2}$	0.45	1.02	2.01	2.01	1.16
$8.3\underline{3}.10^{-2}$	0.45	1.02	3.02	3.02	1.73
$6.25.10^{-2}$	0.45	1.02	4.02	4.02	2.31
$12.50.10^{-2}$	0.04	0	1.01	1.01	0.58
$8.3\underline{3}.10^{-2}$	0.04	0	1.01	2.01	0.58

A difference in convergence behavior of peak and RMS values is noticeable. Convergence of RMS values to the exact solution as $\Delta h \rightarrow 0$ in Table 6-2 seems “faster” than convergence of peak values in Table 6-1. This observation is due to the fact that convergence of a point-wise quantity, that is, a quantity estimated at a single computational cell, is more difficult to achieve than convergence of an average. This remark strengthens the claim that caution should be exercising when formulating point-wise error models such as those of equations (4-9) or (4-20).

6.3 Linear, Space-only Error Ansatz Models for the Vortex Problem

In this section, results are presented of formulating linear, space-only error Ansatz models for studying solution convergence. This approach is consistent with common practices for code verification and solution self-convergence, such as those encountered for the verification of Eulerian or Lagrangian hydro-dynamics [12] or structural mechanics [6] calculations.

When assessing solution convergence error, it is common practice to account for the nature of the functions being solved for by using an appropriate norm of the L^p family. The L^2 norm is the natural choice for continuous functions while L^1 or L^∞ norms are preferred for discontinuous functions. The Euclidean (L^2) norm is used in the remainder because the flow condition for the Vortex2D problem is smooth (non-shocked).

Solution convergence errors are calculated over the computational domain $0 \le X \le 10$ cm and $0 \le Y \le 10$ cm with the L^1, L^2, and L^∞ norms. Values are reported in Table 6-3 for the density (ρ), X-momentum (ρU_x), Y-momentum (ρU_y), and total energy (E) state variables. Being defined as an average, the L^1 and L^2 norms are scaled as ($1/N_{Cells}$) to avoid biasing the results by the number of cells used in the computational grid:

$$\begin{aligned}
\left\| y^* - y_{h,t} \right\|_{L^1} &= \frac{1}{N_{Cells}} \sum_{k=1\cdots N_{Cells}} \left| y^*(k) - y_{h,t}(k) \right| \\
\left\| y^* - y_{h,t} \right\|_{L^2} &= \sqrt{\frac{1}{N_{Cells}} \sum_{k=1\cdots N_{Cells}} \left(y^*(k) - y_{h,t}(k) \right)^2} \\
\left\| y^* - y_{h,t} \right\|_{L^\infty} &= \max_{k=1\cdots N_{Cells}} \left| y^*(k) - y_{h,t}(k) \right|
\end{aligned} \tag{6-8}$$

Note that, being a point-wise quantity estimated on the single computational cell that returns the maximum error, the L^∞ norm does not need to be scaled like the L^1 and L^2 norms are.

The same trends seen in Tables 6-1 and 6-2 are also visible in Table 6-3. First, the three L^p norms indicate convergence as the computational grid is refined, $\Delta h \rightarrow 0$, for all state variables reported in Table 6-3. Second, running the calculation at a lower CFL number tends to increase the solution error, hence, suggesting a time effect $(\Delta t)^q$ where the exponent is negative, $q < 0$.

Values obtained with the L^2 norm of solution errors for the total energy field (E) are shown graphically in Figures 6-2 and 6-3. The three curves in Figure 6-2 correspond to different CFL conditions. The figure shows convergence as a function of cell size. Since the curves obtained with smaller CFL numbers are "above" those obtained with larger CFL numbers, Figure 6-2 also indicates that solution convergence errors increase as $\Delta t \rightarrow 0$. Discontinuity in the slope of curves obtained at $N_{CFL} = 0.90$ and $N_{CFL} = 0.45$ are observed because the refinement ratios are different, as previously noted, between the coarse-to-medium grids and medium-to-fine grids. Visually, the convergence with spatial discretization, $(\Delta h)^p$, occurs at a rate estimated between one and two, $1 \le p \le 2$, because the solution error decreases as $(R_H)^1$ to $(R_H)^2$.

The three error curves in Figure 6-3 correspond to different computational grids. The figure shows convergence as a function of CFL condition. Visual observation of Figure 6-3 indicates that convergence with the CFL condition, $(N_{CFL})^q$ or $(\Delta t)^q$, occurs at a rate between minus one and minus two, $-1 \le q \le -2$, because the solution error decreases as $(R_H)^{-2}$ to $(R_H)^{-1}$. Negative exponents are consistent with the fact that the solution error is inversely proportional to N_{CFL}.

Table 6-3. L^1, L^2, and L^∞ norms of the numerical solution error (Vortex2D).

Run Settings		L^1 Norms of Solution Convergence Error			
Cell Size, Δh (cm)	**N_{CFL} (Unitless)**	**Density, ρ (gm-cm^{-3})**	**Momentum, ρU_x (gm-cm^{-2}-sec^{-1})**	**Momentum, ρU_y (gm-cm^{-2}-sec^{-1})**	**Energy, E (erg-gm^{-1})**
$12.50.10^{-2}$	0.90	$0.18.10^{-2}$	$0.44.10^{-2}$	$0.54.10^{-2}$	$1.04.10^{-2}$
$8.3\underline{3}.10^{-2}$	0.90	$0.35.10^{-2}$	$0.90.10^{-2}$	$1.18.10^{-2}$	$2.11.10^{-2}$
$6.25.10^{-2}$	0.90	$0.58.10^{-2}$	$1.53.10^{-2}$	$2.01.10^{-2}$	$3.57.10^{-2}$
$12.50.10^{-2}$	0.45	$0.07.10^{-2}$	$0.17.10^{-2}$	$0.22.10^{-2}$	$0.39.10^{-2}$
$8.3\underline{3}.10^{-2}$	0.45	$0.15.10^{-2}$	$0.38.10^{-2}$	$0.52.10^{-2}$	$0.90.10^{-2}$
$6.25.10^{-2}$	0.45	$0.26.10^{-2}$	$0.70.10^{-2}$	$0.94.10^{-2}$	$1.64.10^{-2}$
$12.50.10^{-2}$	0.04	$0.03.10^{-2}$	$0.09.10^{-2}$	$0.12.10^{-2}$	$0.20.10^{-2}$
$8.3\underline{3}.10^{-2}$	0.04	$0.08.10^{-2}$	$0.21.10^{-2}$	$0.29.10^{-2}$	$0.49.10^{-2}$

Run Settings		L^2 Norms of Solution Convergence Error			
Cell Size, Δh (cm)	**N_{CFL} (Unitless)**	**Density, ρ (gm-cm^{-3})**	**Momentum, ρU_x (gm-cm^{-2}-sec^{-1})**	**Momentum, ρU_y (gm-cm^{-2}-sec^{-1})**	**Energy, E (erg-gm^{-1})**
$12.50.10^{-2}$	0.90	$0.60.10^{-2}$	$1.15.10^{-2}$	$1.47.10^{-2}$	$3.28.10^{-2}$
$8.3\underline{3}.10^{-2}$	0.90	$1.21.10^{-2}$	$2.31.10^{-2}$	$3.23.10^{-2}$	$6.71.10^{-2}$
$6.25.10^{-2}$	0.90	$1.98.10^{-2}$	$3.84.10^{-2}$	$5.39.10^{-2}$	$10.98.10^{-2}$
$12.50.10^{-2}$	0.45	$0.23.10^{-2}$	$0.45.10^{-2}$	$0.61.10^{-2}$	$1.29.10^{-2}$
$8.3\underline{3}.10^{-2}$	0.45	$0.54.10^{-2}$	$1.00.10^{-2}$	$1.45.10^{-2}$	$2.94.10^{-2}$
$6.25.10^{-2}$	0.45	$0.97.10^{-2}$	$1.80.10^{-2}$	$2.64.10^{-2}$	$5.27.10^{-2}$
$12.50.10^{-2}$	0.04	$0.12.10^{-2}$	$0.24.10^{-2}$	$0.33.10^{-2}$	$0.68.10^{-2}$
$8.3\underline{3}.10^{-2}$	0.04	$0.30.10^{-2}$	$0.55.10^{-2}$	$0.81.10^{-2}$	$1.62.10^{-2}$

Run Settings		L^∞ Norms of Solution Convergence Error			
Cell Size, Δh (cm)	**N_{CFL} (Unitless)**	**Density, ρ (gm-cm^{-3})**	**Momentum, ρU_x (gm-cm^{-2}-sec^{-1})**	**Momentum, ρU_y (gm-cm^{-2}-sec^{-1})**	**Energy, E (erg-gm^{-1})**
$12.50.10^{-2}$	0.90	$8.29.10^{-2}$	$11.34.10^{-2}$	$16.04.10^{-2}$	$48.34.10^{-2}$
$8.3\underline{3}.10^{-2}$	0.90	$16.19.10^{-2}$	$20.65.10^{-2}$	$34.17.10^{-2}$	$95.02.10^{-2}$
$6.25.10^{-2}$	0.90	$24.47.10^{-2}$	$32.21.10^{-2}$	$53.86.10^{-2}$	$145.92.10^{-2}$
$12.50.10^{-2}$	0.45	$3.37.10^{-2}$	$4.83.10^{-2}$	$7.32.10^{-2}$	$20.41.10^{-2}$
$8.3\underline{3}.10^{-2}$	0.45	$7.26.10^{-2}$	$9.29.10^{-2}$	$16.66.10^{-2}$	$44.38.10^{-2}$
$6.25.10^{-2}$	0.45	$12.48.10^{-2}$	$15.56.10^{-2}$	$29.19.10^{-2}$	$76.32.10^{-2}$
$12.50.10^{-2}$	0.04	$1.74.10^{-2}$	$2.62.10^{-2}$	$3.97.10^{-2}$	$10.82.10^{-2}$
$8.3\underline{3}.10^{-2}$	0.04	0	$1.01.10^{-2}$	$2.01.10^{-2}$	$0.58.10^{-2}$

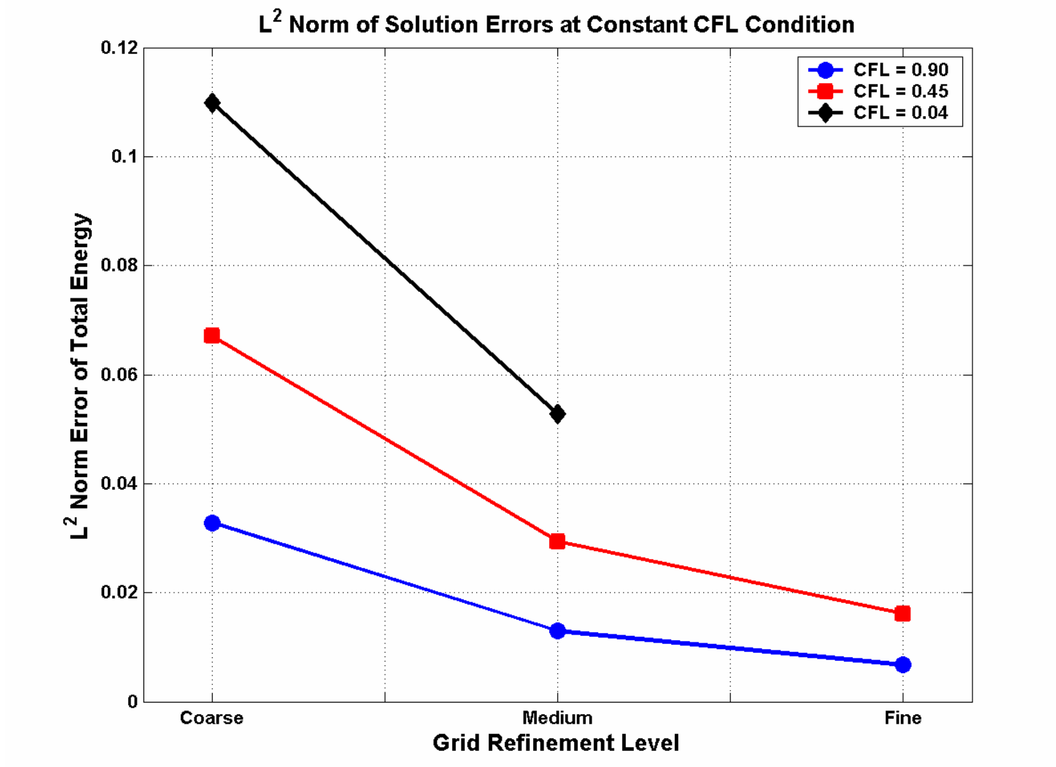


Figure 6-2. Convergence of the total energy field as a function of Δh (Vortex2D).

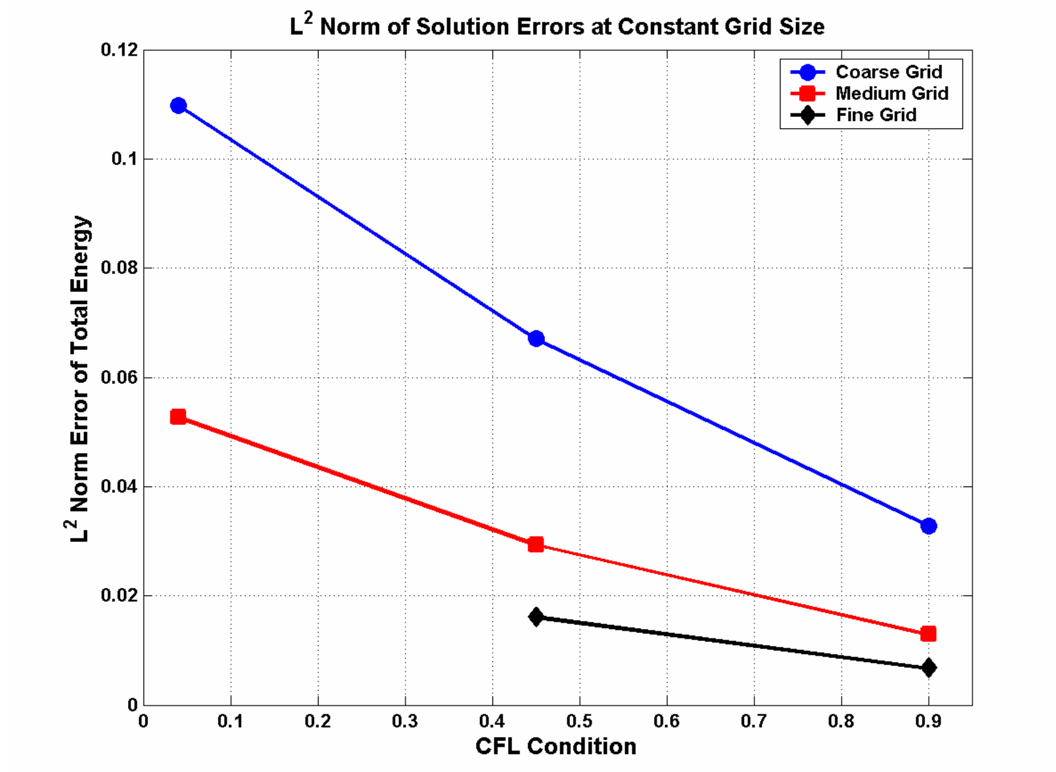


Figure 6-3. Convergence of the total energy field as a function of N_{CFL} (Vortex2D).

The next step of the analysis is to use the data listed in Table 6-3 to best-fit a spatial error Ansatz model that takes the following functional form:

$$e_h = \left\| y^* - y_{h,t} \right\|_{L^p} = \beta_o + \beta_1 (\Delta h)^p + H.O.T. \tag{6-9}$$

Equation (6-9) restricts the analysis to monotonic convergence behavior and it assumes that the CFL condition and, therefore, the time step, have no significant effect on the prediction of solution error. This latter assumption is clearly contradicted by Figures 6-2 and 6-3.

A non-linear optimization solver is used to fit the unknown parameters (β_o;β_1;p) of the error Ansatz model. The cost function for numerical optimization is defined as:

$$C = 1 - e^{-MSE}, \quad MSE = \sqrt{\frac{1}{N_{Runs}} \sum_{k=1 \cdots N_{Runs}} \left(e_h(k) - \hat{e}_h(k) \right)^2} \tag{6-10}$$

where e_h denotes the solution error values listed in Table 6-3; $\hat{e}_h$ represents predictions of the error Ansatz equation (6-9); N_{Runs} is the number of available computer runs (here, N_{Runs} = 8); and “k” identifies the computer run, k = 1 … N_{Runs}. The exponential function serves the purpose of accentuating the “shape” of the Mean Square Error (MSE), which speeds-up convergence.

For a given combination of model parameters (β_o;β_1;p), the solution errors $\hat{e}_h$ are estimated using equation (6-9), then, the goodness-of-fit of this particular model is assessed with equation (6-10). The numerical optimization solver uses this information to calculate gradients and to search for the triplet (β_o;β_1;p) of model parameters that minimize the cost function. Convergence criteria are defined to stop the optimization iterations when model parameters are not modified by more than 10^{-6} or the cost function has decreased by a factor of 10^{-3} compared to its initial value. The algorithm used is the Simplex algorithm of MATLAB™ (named “`fminsearch`”) that implements a gradient-based search of the steepest descent.

It is also verified that a least-squares solver provides the same solution as the non-linear optimization solver. The least-squares solver handles over-determined systems, that is, the fact that 8 equations are available to fit only 3 parameters, using a Singular Value Decomposition (SVD) algorithm that eliminates any singularity or ill-conditioning of the effect matrix, therefore, making it possible to invert it.

Table 6-4. Parameters of linear, spatial error Ansatz models (Vortex2D).

Estimate of Model Parameter	**State Variable Response**			
	Density, ρ ($gm\text{-}cm^{-3}$)	**Momentum, ρU_x ($gm\text{-}cm^{-2}\text{-}sec^{-1}$)**	**Momentum, ρU_y ($gm\text{-}cm^{-2}\text{-}sec^{-1}$)**	**Energy, E ($erg\text{-}gm^{-1}$)**
Exponent, p	1.9882	1.7703	1.6261	1.2915
Intercept, β_o	-0.0008	-0.0036	-0.0062	-0.0273
Slope, β_1	0.8456	1.1182	1.1815	1.4326

Solutions (β_o;β_1;p) are listed in Table 6-4 for the four state variables density (ρ), X-momentum (ρU_x), Y-momentum (ρU_y), and total energy (E) when the L^2 norm metric is used to define solution convergence errors. The exponent (p) is dimensionless. The intercept coefficient (β_o) is given in units of EU and the slope coefficient (β_1) is given in units of EU-cm^{-p}, where the Engineering Units (EU) used are specific to the state variable considered. For example, EU = $gm\text{-}cm^{-3}$ for the density field.

Two observations are made. First, the intercept coefficients are more than two orders of magnitude smaller than the slope coefficients, except for the energy state variable. The significance of the intercept coefficient is that, if the numerical method under consideration is consistent, then the 0th-order error should be zero. The small values obtained indicate consistency, that is, the numerical solution converges to the continuous solution as $\Delta h \rightarrow 0$. The second observation is that, with the exception of the energy field, values of the exponent indicate near-to-quadratic convergence rates. This result matches the theoretical order of spatial convergence (p = 2) for the hydro-code considered in the presence of a smooth flow.

6.4 Non-linear, Space-time Error Ansatz Models for the Vortex Problem

To assess, first, the effect of time discretization on solution convergence and, second, the degree of to which spatial and temporal discretization effects may be coupled, a non-linear error Ansatz model is investigated:

$$e_h = \left\| y^* - y_{h,t} \right\|_{L^p} = \beta_o + \beta_1 (\Delta h)^p + \beta_2 (\Delta t)^q + \beta_3 (\Delta h)^r (\Delta t)^s + \text{H.O.T.} \quad (6\text{-}11)$$

Data sets from Table 6-3 are used, as before, to fit the unknown parameters ($\beta_o;\beta_1;\beta_2;\beta_3;p;q;r;s$) of equation (6-11). The results reported below are obtained by using the L^2 norm of solution errors for the four state variables density, x-momentum, y-momentum, and total energy.

To develop an error Ansatz equation that is a function of the time step (Δt), as opposed to the CFL condition (N_{CFL}), a characteristic time scale is sought for each run. Time step statistics shown in Table 6-5 are obtained by examining the output of the code that has the capability to dump the increment Δt used at each cycle of the time integration scheme. It is observed that the time steps selected during the simulation do not vary significantly (less than 10% variation) and that the maximum values are very close to the mean values. The maximum time step (Δt_{Max}) is used to define a characteristic time scale for each run because a general principle for solving hyperbolic systems of PDE is to select a time step closest to the stability limit (CFL condition). This helps to keep the numerical dispersion and dissipation small.

Table 6-5. Time steps observed during the simulation runs (Vortex2D).

Run Number	Cell Size, Δh (cm)	N_{CFL} (Unitless)	Δt_{Mean} (sec.)	Δt_{Max} (sec.)	$\sigma(\Delta t)$ (sec.)	Coefficient of Variation, $\sigma(\Delta t)/\Delta t_{Mean}$
1	$12.50.10^{-2}$	0.90	0.0385	0.0396	0.0035	8.99%
2	$8.3\underline{3}.10^{-2}$	0.90	0.0195	0.0200	0.0013	6.92%
3	$6.25.10^{-2}$	0.90	0.0010	0.0010	0.0000	1.58%
4	$12.50.10^{-2}$	0.45	0.0254	0.0259	0.0024	9.56%
5	$8.3\underline{3}.10^{-2}$	0.45	0.0128	0.0130	0.0009	6.78%
6	$6.25.10^{-2}$	0.45	0.0006	0.0006	0.0000	0.87%
7	$12.50.10^{-2}$	0.04	0.0191	0.0193	0.0015	8.06%
8	$8.3\underline{3}.10^{-2}$	0.04	0.0096	0.0097	0.0005	5.00%

The design of computer experiments used for this study consists of the eight runs shown in Table 6-5 characterized by two input factors, the cell size (Δh) and the maximum time step (Δt_{Max}). Error Ansatz models are formulated for the L^2 norm of solution error.

The statistical effect screening and model fitting algorithm described in Section 5 is applied to analyze the family of error Ansatz models defined by equation (6-11). The Gibbs sampler that randomly walks the family of models is initialized with 50 iterations for burn-in and 200 iterations for constructing the Markov Chain Monte Carlo (MCMC) sample from which the posterior probability of each effect is estimated. The difference between the first 50 iterations (burn-in) and the next 200 (random walk) is that models visited during the former are disregarded in an attempt to reduce the influence of the starting point on the final marginal probabilities of effects. The prior probabilities are initialized at 25% for the main effects; 10% for a linear interaction that includes an effect also selected as main effect; and 1% only for a linear interaction that includes no effect selected as main effect. The goodness-of-fit of each model visited during the random walk is assessed using the MSE statistic defined in equation (6-10). As explained in Section 5, models that are capable of fitting the data better tend to be visited more often. At the end of the

200 iterations, the posterior probabilities of effects are estimated by counting how many times they appear in the most-often visited models. Details of the procedure are provided in Section 5.

Since the algorithm has the capability to explore linear interactions as well as main effects, this is taken advantage of by defining the family of models as follows:

$$e_h = \left\| y^* - y_{h,t} \right\|_{L^p} = \beta_o + + \beta_1 A + \beta_2 B + \beta_3 C + \beta_4 AB + \beta_5 AC + \beta_6 BC + \varepsilon_R \quad (6\text{-}12)$$

where the residual fitting error $\varepsilon_R \sim N(0;\sigma_R)$ is assumed to be a Gaussian-distributed random noise of unknown variance. The three main effects, labeled "A", "B", and "C" in equation (6-12) represent the spatial and temporal discretization effects considered. Because nothing prevents these effects from depending on one another, they are initialized as A = $(\Delta h)^p$, B = $(\Delta t_{Max})^q$, and C = $(\Delta h)^r(\Delta t_{Max})^s$. Substitution into equation (6-12) gives the functional form of the family of error Ansatz models visited by the MCMC algorithm:

$$\begin{aligned} e_h = \left\| y^* - y_{h,t} \right\|_{L^p} &= \beta_o + + \beta_1 (\Delta h)^p + \beta_2 (\Delta t_{Max})^q + \beta_3 (\Delta h)^r (\Delta t_{Max})^s \\ &+ \beta_4 (\Delta h)^p (\Delta t_{Max})^q + \beta_5 (\Delta h)^{p+r} (\Delta t_{Max})^s + \beta_6 (\Delta h)^r (\Delta t_{Max})^{q+s} + \varepsilon_R \end{aligned} \quad (6\text{-}13)$$

Including the linear interaction terms ("AB", "AC", and "BC") provides a "sanity check" that the basic functional form of the error Ansatz equation (6-11) is correct. If equation (6-11) captures the phenomenon correctly, then it is reasonable to expect that none of the interaction terms shown on the second line of equation (6-13) will come up with a significant posterior probability.

Table 6-6. Parameters of non-linear, space-time error Ansatz models (Vortex2D).

Estimate of Model Parameter	**State Variable Response**			
	Density, ρ ($gm\text{-}cm^{-3}$)	**Momentum, ρU_x ($gm\text{-}cm^{-2}\text{-}sec^{-1}$)**	**Momentum, ρU_y ($gm\text{-}cm^{-2}\text{-}sec^{-1}$)**	**Energy, E ($erg\text{-}gm^{-1}$)**
Exponent, p	2.151	1.779	1.574	1.129
Exponent, q	-0.978	-1.212	-1.092	-1.461
Exponent, r	2.068	1.934	1.841	1.865
Exponent, s	-1.209	-1.169	-1.207	-1.224
Intercept, β_o	$7.448.10^{-3}$	$14.170.10^{-3}$	$19.920.10^{-3}$	$40.960.10^{-3}$
Coefficient, β_1	$2.251.10^{-3}$	$5.896.10^{-3}$	$5.365.10^{-3}$	$16.690.10^{-3}$
Coefficient, β_2	$-0.208.10^{-3}$	$0.194.10^{-3}$	$-0.690.10^{-3}$	$0.391.10^{-3}$
Coefficient, β_3	$4.533.10^{-3}$	$4.965.10^{-3}$	$10.780.10^{-3}$	$14.820.10^{-3}$
Coefficient, β_4	$-0.276.10^{-3}$	$0.333.10^{-3}$	$-0.431.10^{-3}$	$-0.933.10^{-3}$
Coefficient, β_5	$0.433.10^{-3}$	$0.366.10^{-3}$	$-0.092.10^{-3}$	$-0.769.10^{-3}$
Coefficient, β_6	$-0.027.10^{-3}$	$0.341.10^{-3}$	$-0.023.10^{-3}$	$1.782.10^{-3}$

The results of Bayesian model screening are summarized in Tables 6-6 and 6-7. Table 6-6 lists the model parameters ($\beta_o;\beta_1;\beta_2;\beta_3;p;q;r;s$) that correspond to the most-often visited solution error Ansatz models for each of the state variables considered. Also listed are the coefficients ($\beta_4;\beta_5;\beta_6$) added to study the significance of interaction terms in equation (6-13). Like for Table 6-4, the exponents (p;q;r;s) are dimensionless. The regression coefficients (β_k) are given in units of EU-cm^{-a}-sec^{-b} where "a" and "b" denote the space and time exponents of the corresponding effect, and EU denotes the engineering unit of the state variable considered. For example, the regression coefficient β_3 is listed in units of EU-cm^{-r}-$sec.^{-s}$ in Table 6-6.

Overall, it can be observed from Table 6-6 that the rates of convergence are found to be between one and two ($1 \le p \le 2$) for the effect $(\Delta h)^p$ of spatial discretization; q = -1 for the effect $(\Delta t_{Max})^q$ of temporal discretization; and (r;s) = (2;-1) for the non-linear coupling effect between space and time $(\Delta h)^r(\Delta t_{Max})^s$. These results do not contradict those of the space-only error analysis documented in Table 6-4. It is interesting to note that the coupling term $(\Delta h)^2(\Delta t_{Max})^{-1}$

can be viewed as a linear interaction between the space discretization effect and the cell size-to-time step ratio, that is, $(\Delta h)^1(V_{Cell})^1$ where $V_{Cell} = (\Delta h/\Delta t)$ would be a new explanatory variable that represents the speed at which a cell can propagate information to its neighbors.

Table 6-7 lists the prior and posterior probabilities of each effect. The effects investigated include those of the generic error Ansatz model (6-11) together with the space-time interactions added on the second line of equation (6-13). Effect screening consists of jointly examining the Tables 6-6 and 6-7 to identify those effects that provide high posterior probabilities as well as large values of the regression coefficient.

Table 6-7. Prior and posterior probabilities for effect screening (Vortex2D).

Effect Considered	Density (ρ)		X-momentum (ρU_x)		Y-momentum (ρU_y)		Energy (E)	
	Prior	Posterior	Prior	Posterior	Prior	Posterior	Prior	Posterior
$\beta_1(\Delta h)^p$	25%	78%	25%	99%	25%	66%	25%	100%
$\beta_2(\Delta t_{Max})^q$	25%	17%	25%	28%	25%	18%	25%	34%
$\beta_3(\Delta h)^r(\Delta t_{Max})^s$	25%	49%	25%	67%	25%	72%	25%	60%
$\beta_4(\Delta h)^p(\Delta t_{Max})^q$	10%	9%	10%	6%	10%	15%	10%	6%
$\beta_5(\Delta h)^{p+r}(\Delta t_{Max})^s$	10%	7%	10%	31%	10%	13%	10%	23%
$\beta_6(\Delta h)^r(\Delta t_{Max})^{q+s}$	10%	9%	10%	13%	10%	18%	10%	7%

Based on examining the increase in posterior probability (or lack thereof), the algorithm suggests to restrict the error Ansatz models to the space discretization effect $(\Delta h)^p$ and space-time coupling $(\Delta h)^r(\Delta t_{Max})^s$ for all state variables considered. The other effects are eliminated without significantly degrading the goodness-of-fit. The resulting error Ansatz equation is:

$$e_h = \left\| y^* - y_{h,t} \right\|_{L^p} = \beta_o + \beta_1 (\Delta h)^p + \beta_3 (\Delta h)^r (\Delta t)^s \quad (6\text{-}14)$$

with exponents and coefficients listed in Table 6-6 for each state variable. It is emphasized that this equation applies only to the prediction of L^2 error norms, as defined in expression (6-8).

For the density (ρ), the best model (6-14) is visited 65 times in 200 iterations and its MSE is equal to 20.5%. For the momentum in the X-direction (ρU_x), the best model (6-14) is visited 55 times in 200 iterations and its MSE is equal to 19.3%. For the momentum in the Y-direction (ρU_y), the best model (6-14) is visited 57 times in 200 iterations and its MSE is equal to 21.0%. For the total energy (E), the best model (6-14) is visited 53 times in 200 iterations and its MSE is equal to 18.8%.

The fact that the interaction terms added on the second line of equation (6-13) are rejected is good news. It means that the optimization search does not need to include them to improve the goodness-of-fit. This is interpreted as evidence that the functional form of the space-time error Ansatz model proposed in equation (6-11) is correct. A color code is shown in Table 6-7 to summarize the results. The significant effects are shown in **green**, the non-significant effects are shown in **red**, and the effects that it might be prudent to include are shown in **gold**.

This page is left blank intentionally.

7. Results Obtained With Four Code Verification Test Problems

Summary: The results of fitting space-only and space-time models of solution convergence error are illustrated with four code verification test problems. The test problems are selected to represent a combination of convergent, divergent, or periodic flows; linear or non-linear equations; and smooth or shocked solutions. Another criterion of definition of test problems is that an analytical solution of the equations is available in closed form. The test problems are based on the Euler equations of compressible gas dynamics, solved either in one dimension (spherical) or in two dimensions. The Noh problem represents the non-linear and shocked compression of an ideal gas. The Sedov problem is a non-linear and shocked expansion of a blast-wave. The vortex evolution problem provides a non-linear but smooth solution. Finally, the wave propagation problem provides a linear and smooth solution. Results obtained with the four test problems indicate, first, that the functional form of the model matters greatly and, second, that time effects are almost always found to be significant.

The results obtained are illustrated with four code verification test problems. The test problems have in common that they all solve the Euler equations of compressible gas dynamics in one dimension (1D) or two dimensions (2D). Where they differ is that they represent convergent (Noh), divergent (Sedov), or periodic (Vortex, Wave) flow conditions. Their solutions are either shocked (Noh, Sedov) or smooth (Vortex, Wave). These characteristics of test problems are summarized in Table 7-1. Table 7-2 lists the symbols and notations used.

Table 7-1. Characteristics of the four code verification test problems.

Test Problem	Symbol	Dimension	Type of Equations		Type of Solutions	
			Linear	Non-linear	Smooth	Shocked
Noh	Noh1D	1D		•		•
Noh	Noh2D	2D		•		•
Sedov	Sedov1D	1D		•		•
Sedov	Sedov2D	2D		•		•
Vortex Evolution	Vortex2D	2D		•	•	
Wave Propagation	Wave2D	2D	•		•	

Test Problem	Symbol	Dimension	Flow Condition		
			Convergent	Divergent	Periodic
Noh	Noh1D	1D	•		
Noh	Noh2D	2D	•		
Sedov	Sedov1D	1D		•	
Sedov	Sedov2D	2D		•	
Vortex Evolution	Vortex2D	2D			•
Wave Propagation	Wave2D	2D			•

Table 7-2. Symbols and notations used in Section 7.

Symbol	Description	Units
ρ	Density	g-cm^{-3}
p	Pressure	dyne-cm^{-2} = 10^{-1} Pa (N-m^{-2})
U_x, U_y	Flow velocity	cm-sec.$^{-1}$
E	Total energy	erg-gm^{-1}
e	Specific Internal Energy (SIE)	erg-gm^{-1}
T	Temperature	eV
y*	Exact solution of the continuous equations	Engineering Units (EU)
$y_{h,t}$	Numerical solution of the discretized equations	Engineering Units (EU)
e_h	Solution error, $e_h = \| y^* - y_{h,t} \|$	Engineering Units (EU)
Δh	Cell or element size	cm
Δt	Time step	sec.
p, q, r, s	Exponents of the solution error model	None
β_k	Regression coefficients of the error model	Engineering Units (EU)
$(\ )_C$, $(\ )_M$, $(\ )_F$	Subscripts for coarse, medium, and fine resolutions	N/A

The hydro-code RAGE developed under the Los Alamos Code Project "Crestone" is used to solve the four code verification test problems considered. RAGE is an Adaptive Mesh Refinement (AMR) Eulerian hydro-dynamics code that uses a Riemann solver-based Godunov method to obtain the solution of the gas dynamics equations [2]. RAGE release 2005.03.31.000 is used on the Los Alamos platform QSC for this work. Cell refinement is triggered by either a high degree of cell compression or the detection of large gradients in the density or pressure fields. The time stepping strategy implemented in RAGE for AMR grids is the so-called "locked" approach where all cells, refined or not, use the same (smallest) time step to advance their solution in time. Only 1D or 2D Cartesian geometries are considered in the test problems. Even though the methodology proposed is general-purpose, discretization along different dimensions is constrained to be equivalent, $\Delta h_x = \Delta h_y$. With minor modification, the error Ansatz models may be extended to other multi-dimension geometries or discretization schemes without equivalent spacing.

Results of the analyses of the four code verification test problems are discussed below. Each time the test problem is introduced by briefly discussing its setting and characteristics of the analysis performed. Then, the main results are presented by focusing on comparing the solution convergence error Ansatz models formulated in space only to those that include time-space coupling terms. The Noh problem is discussed in Section 7.1, followed by the Sedov problem (Section 7.2), the vortex evolution problem (Section 7.3) and, finally, the wave propagation problem (Section 7.4). A brief description of the toolbox created to generate these results is included in Section 7-5.

7.1 Results Obtained With the Noh Test Problem

The Noh problem represents the shocked compression of an ideal gas. The non-linearity of Euler equations is exercised and the solution is non-smooth (discontinuous). An initial density of ρ = 1.0 gm-cm^{-3} is directed towards the origin at the flow velocity U_x = -1.0 cm-sec^{-1}. It results in an infinite strength circularly symmetric shock reflecting from the origin.[17] References [19] and [20] describe the setting of the Noh problem and its analytic solution, available in a reference

[17] See Reference [19] and, more recently, Reference [20].

frame relative to the shock position. The Noh problem tests the ability of the code to convert kinetic energy into internal energy.

The Euler equations for the Noh problem are solved in one dimension with a spherical symmetry. The design of computer experiments is a three-level full-factorial array totaling $3^2 = 9$ runs with cell sizes $\Delta h = (10^{-1} \mid 10^{-2} \mid 10^{-3})$ cm and time steps $\Delta t = (10^{-2} \mid 10^{-3} \mid 10^{-4})$ sec. The refinement ratios are therefore $R_H = 10$ in space and $R_T = 10$ in time. Results are reported at the final simulation time of 0.6 sec. Figure 7-1 illustrates the full-factorial design of experiments where star symbols represent the nine settings ($\Delta h;\Delta t$) of cell sizes and time steps at which the computer code is analyzed. Pairs ($\Delta h;\Delta t$) illustrated in Figure 7-1 are listed in the table below:

Run Number	Cell Size, Δh (cm)	Time Step, Δt (sec.)		Run Number	Cell Size, Δh (cm)	Time Step, Δt (sec.)
1	10^{-1}	10^{-2}		**6**	10^{-3}	10^{-3}
2	10^{-2}	10^{-2}		**7**	10^{-1}	10^{-4}
3	10^{-3}	10^{-2}		**8**	10^{-2}	10^{-4}
4	10^{-1}	10^{-3}		**9**	10^{-3}	10^{-4}
5	10^{-2}	10^{-3}				

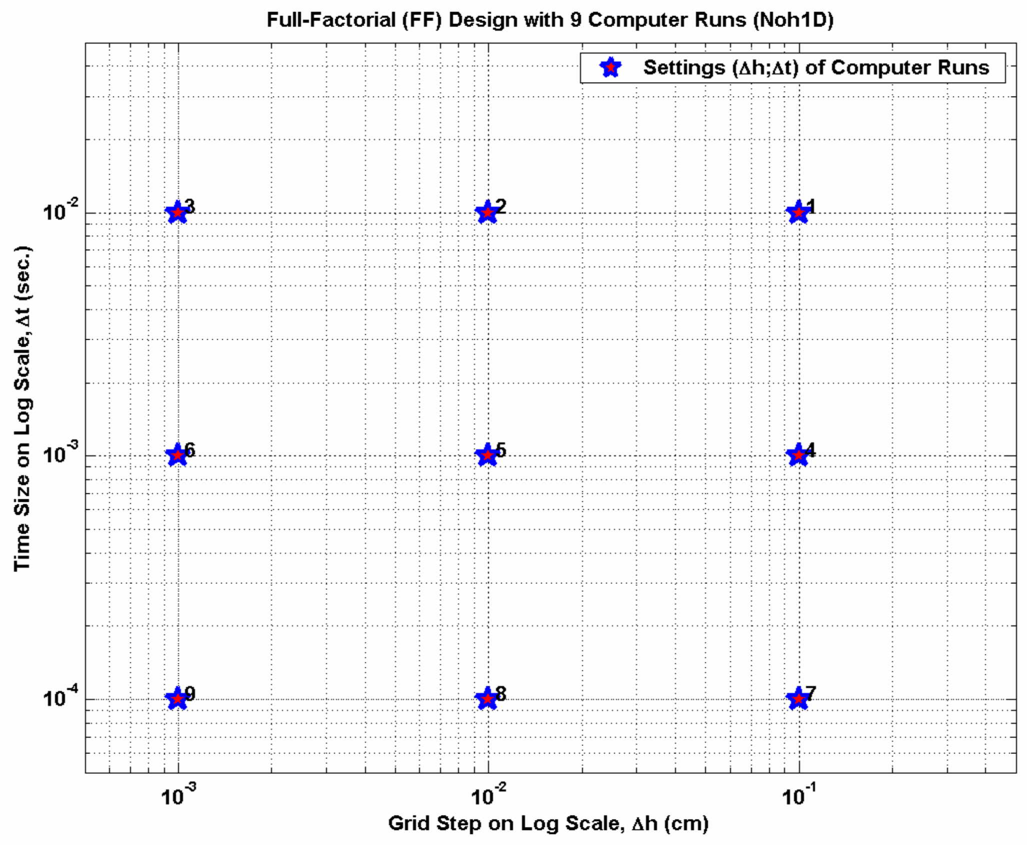


Figure 7-1. Full-factorial design of computer experiments for the Noh1D problem.

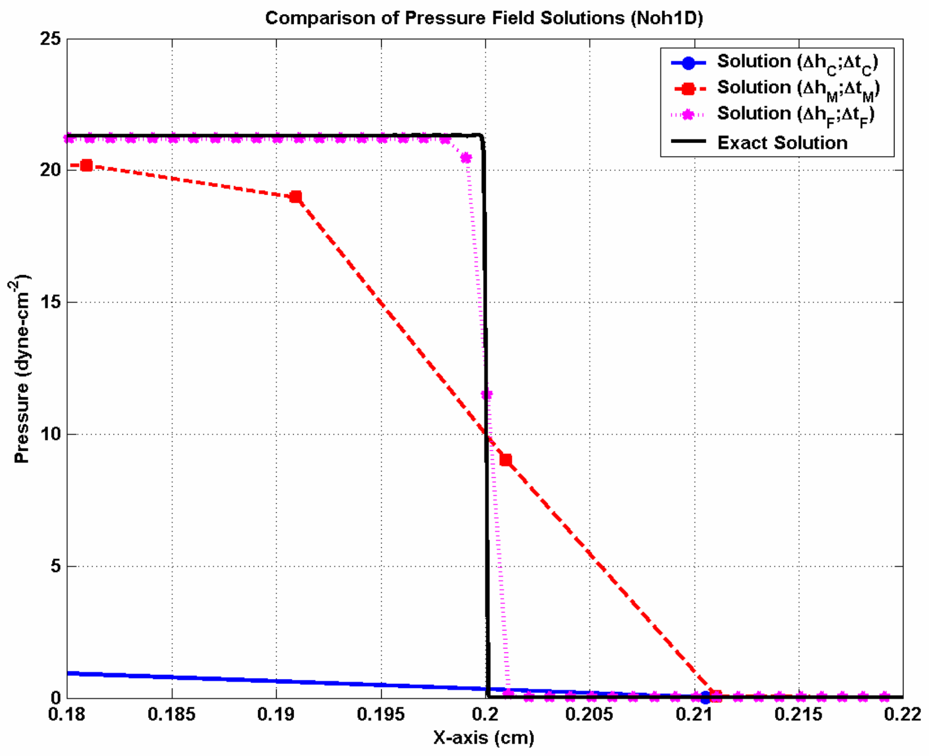


Figure 7-2. Analytical and numerical pressure fields of the Noh1D problem.

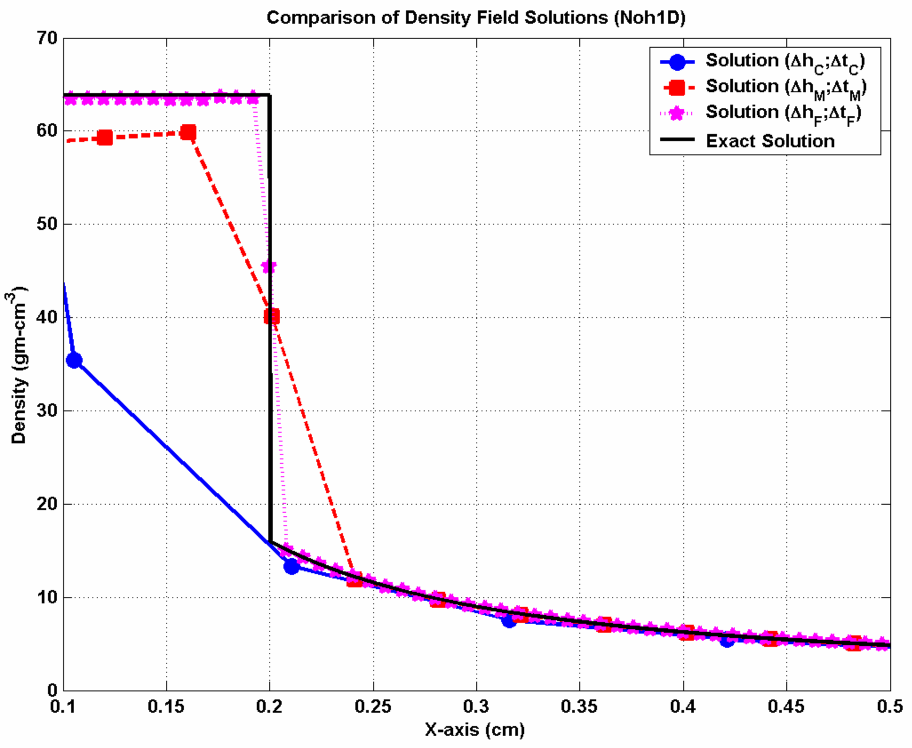


Figure 7-3. Analytical and numerical density fields of the Noh1D problem.

Figure 7-2 illustrates three numerical solutions for the pressure fields obtained with settings ($\Delta h_C;\Delta t_C$), ($\Delta h_M;\Delta t_M$), and ($\Delta h_F;\Delta t_F$). Figure 7-3 shows a similar illustration for the density fields. In both cases, the exact solution is shown and it can be observed that the most refined solution provided with ($\Delta h_F;\Delta t_F$) is in good qualitative agreement with the exact solution. Note that, because the size of the computational domain is [0; 2] cm, Figures 7-2 and 7-3 emphasize a small region near the shock (discontinuity) at coordinate X = 0.2 cm.

Tables 7-3 and 7-4 list the solution convergence error Ansatz models obtained for the pressure and density fields, respectively. The Euclidean L^2 norm is used to calculate the error between numerical and exact solutions over the entire computational domain, that is, $0 \le X \le 2$ cm. Because the numerical solutions are obtained with different numbers of cells (20, 200, and 2,000 cells for the coarse, medium, and fine resolutions) and to avoid biasing the analysis, the L^2 norm of solution error is scaled by the total numbers of cells of each calculation:

$$e_h = \sqrt{\frac{1}{N_{Cells}} \sum_{k=1 \cdots N_{Cells}} \left(y^*(k) - y_{h,t}(k)\right)^2} \tag{7-1}$$

where N_{Cells} denotes the total number of cells used in a calculation and the index “k” is the cell identifier, $1 \le k \le N_{Cells}$. Although not used here, the L^1 norm is always scaled in a similar way. (Being defined as a maximum value, the L^∞ norm is insensitive to the number of cells.)

Table 7-3. Error Ansatz models for the Noh1D problem and pressure field.

Analysis Method	Best Solution Error Model	Mean Square Error (MSE)
Space-only analysis at $\Delta t = 10^{-4}$ sec.	$e_h = 0.29 + 139.51(\Delta h)^{1.14}$	0.0%
Space-only analysis at all time steps	$e_h = 2.08 + 1{,}642.34(\Delta h)^{2.34}$	3.9%
Space-time analysis at all time steps	$e_h = 5.20 + 3.86(\Delta h)^{0.05} - 0.39(\Delta t)^{-0.34} + 2.18(\Delta h)^{0.82}(\Delta t)^{-0.38}$	3.2%
Space-time analysis at all time steps, with $\log_{10}$ scaling	$e_h = 4.29 + 6.53(E_H)^{-15.68} - 0.43(E_T)^{-6.93}$	2.1%

Table 7-4. Error Ansatz models for the Noh1D problem and density field.

Analysis Method	Best Solution Error Model	Mean Square Error (MSE)
Space-only analysis at $\Delta t = 10^{-4}$ sec.	$e_h = 0.85 + 416.46(\Delta h)^{1.13}$	0.0%
Space-only analysis at all time steps	N/A	N/A
Space-time analysis at all time steps	$e_h = 5.78 + 148.34(\Delta h)^{0.76} - 211.15(\Delta t)^{2.17}$	3.2%
Space-time analysis at all time steps, with $\log_{10}$ scaling	N/A	N/A

All regression coefficients (β_k) shown in Tables 7-3 and 7-4 are expressed in engineering units, except in the case of $\log_{10}$ scaling. Log_{10} scaling converts the cell size Δh and time step Δt into variables E_H and E_T such that:

$$\Delta h = 10^{-E_H}, \quad \Delta t = 10^{-E_T} \tag{7-2}$$

Fitting the error values as a function of exponents ($E_H;E_T$) as opposed to the cell size and time step pair ($\Delta h;\Delta t$) tends to improve the goodness-of-fit because the non-linear relationships shown in equation (7-2) are eliminated and the values of variables E_H and E_T have the same order of magnitude, which may not be the case with variables Δh and Δt. It should, however, be remembered that a positive exponent such as $(\Delta h)^p$, where $p \geq 0$, is converted into a negative exponent $(E_H)^q$, where $q \leq 0$. An effect such as $6.53(E_H)^{-15.68}$ shown in Table 7-3 is somewhat counter-intuitive but it does indicate that reducing the cell size tends to reduce the solution error.

The goodness-of-fit of solution error models, that is, their ability to reproduce the observed error data, is assessed with the Mean Square Error (MSE) metric in Tables 7-3 and 7-4. The MSE is a root-mean square error between predictions and observations, scaled by the standard deviation of observed error data:

$$\mathrm{MSE} = \frac{100}{\sigma_e}\sqrt{\sum_{k=1\cdots N_{Data}}\left(e_h(k) - \hat{e}_h(k)\right)^2}\,(\%) \tag{7-3}$$

with the standard deviation (σ_e) and mean ($\bar{e}$) statistics are estimated as usually as:

$$\mathrm{MSE} = \frac{1}{(N_{Data}-1)}\sqrt{\sum_{k=1\cdots N_{Data}}\left(e_h(k) - \bar{e}\right)^2}$$
$$\bar{e} = \frac{1}{N_{Data}}\sum_{k=1\cdots N_{Data}} e_h(k) \tag{7-4}$$

and where $\hat{e}_h$ denotes the prediction of the error Ansatz model; e_h is the actual (observed) error; and N_{Data} is the number of observed errors, that is, the number of computer runs from which solution fields are extracted. A value equal to a few percents (less than, say, 5%) indicates a good-quality fit to the data. A value less than 1% indicates an exceptional goodness-of-fit.

Table 7-3 lists four solution error models for the pressure field. Table 7-4 lists those of the density field. Four analyses are considered in each table. First, a space-only analysis is performed with the three computer runs that correspond to the finest time resolution $\Delta t = \Delta t_F$. The second analysis is also restricted to the spatial effect $(\Delta h)^p$ but this time all computer runs are considered. Because a space-only solution error model has no mechanism by which to account for the time effect, the second analysis is expected to lead a less favorable goodness-of-fit. The third analysis is a space-time model with variables $(\Delta h;\Delta t)$ and the fourth analysis is a space-time model with variables $(E_H;E_T)$. In all cases, MSE values remain small (less than 5%).

It is observed that in both cases of pressure and density fields, the space-only solution error models find a spatial rate of convergence close to $p = 1$, which is expected in the presence of a discontinuous (shocked) solution. When space-time solution error models are fitted to the data, it is observed that time stepping effects are significant for the density field and no coupling between space and time are found. Although somewhat less significant for the pressure field, time stepping effects are found with negative exponents, which indicates that the pressure error tends to grow as $\Delta t \rightarrow 0$. Comparing the two space-time analyses for the pressure field, that is, with and without $\log_{10}$ scaling, sheds light on the true nature of space-time coupling. When the error Ansatz model is developed as a function of variables $(\Delta h;\Delta t)$, the coupling effect $(\Delta h)^r(\Delta t)^s$ is found to be significant. However, its significance vanishes after transformation from variables $(\Delta h;\Delta t)$ to variables $(E_H;E_T)$. This seems to indicate that the space-time coupling is due to the non-linearity shown in equation (7-2) as much as it is a manifestation of the data.

Because space-only solution error models feature a single effect $(\Delta h)^p$, statistical screening becomes irrelevant. Space-only models are therefore best-fitted to the data using non-linear optimization. Results reported in this work are obtained with the Simplex algorithm implemented in MATLAB™. The cost function for optimization is either defined as the MSE metric of equation (7-3) or as the function $(1 - e^{-MSE})$, whichever provides better convergence. The maximum number of optimization iterations is equal to 100 x N_P where N_P denotes the number of unknown parameters. For example, $N_P = 3$ for unknowns $(\beta_o;\beta_1;p)$ of the error model $e_h = \beta_o + \beta_1(\Delta h)^p$. Tolerances for convergence of the cost function and solution are set to 10^{-6}.

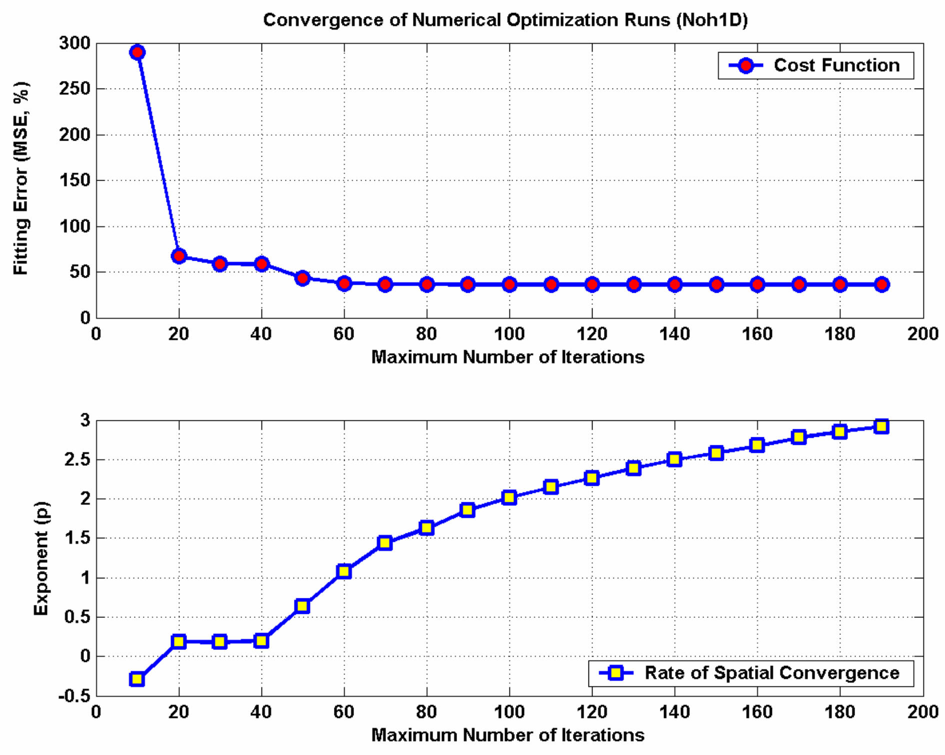


Figure 7-4. Cost function and convergence rate values for the model $e_h = \beta_o + \beta_1(\Delta h)^p$.

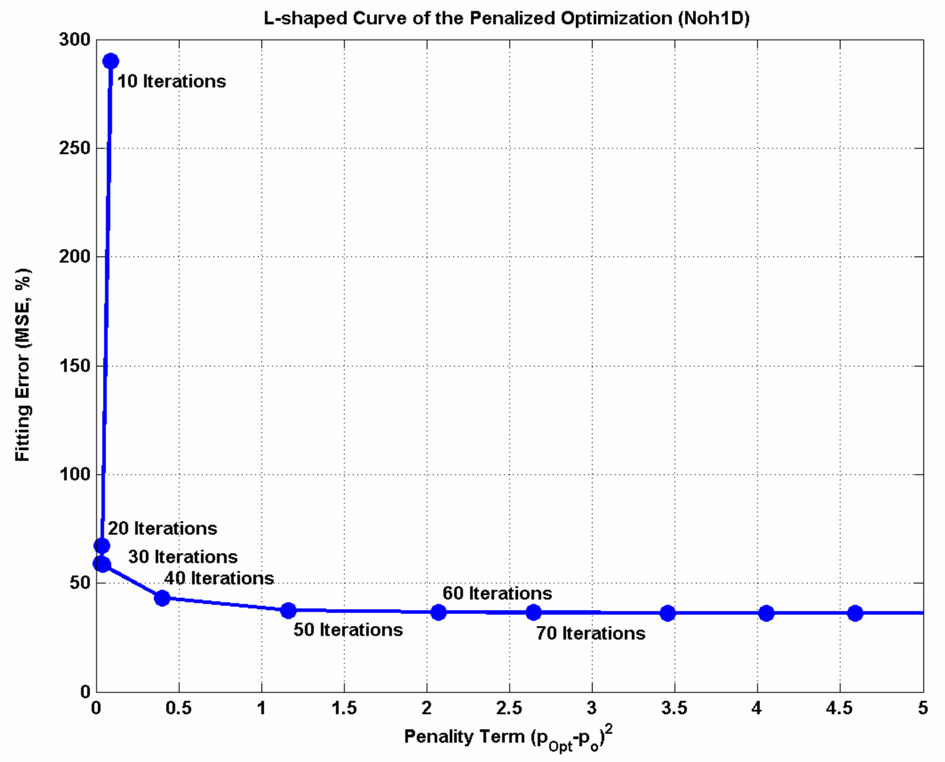


Figure 7-5. L-shaped curve for the optimization of the model $e_h = \beta_o + \beta_1(\Delta h)^p$.

Space-time solution error models are obtained with the Markov Chain Monte Carlo (MCMC) search described in Section 5, which provides not only a best model but also a list of posterior probabilities that indicate whether or not specific effects are important to predict the value of solution error. The settings are similar to those used in Section 6. The total number of MCMC iterations is set to 1,000 x N_P with 100 iterations reserved for burn-in. Models visited during these first 100 burn-in iterations are disregarded to avoid producing posterior statistics that depend on the starting point. Prior probabilities are set to 25% for all main effects; 10% for interactions that involve an effect also selected as main effect; and 1% for linear interactions that do not involve effects selected as main effect. Posterior probabilities are not reported in Tables 7-3, 7-4 (and the following ones) because they are equal to 100%. Effects for which posterior probabilities are not significant are simply not included in the solution error models. An example is the space-time model shown in Table 7-4, $e_h = 5.78 + 148.34(\Delta h)^{0.76} - 211.15(\Delta t)^{2.17}$, where the coupling between space and time is not found significant.

Numerical optimization to best-fit parameters $(\beta_o;\beta_1;p)$ of space-only solution error models $e_h = \beta_o + \beta_1(\Delta h)^p$ is sometimes problematic. Because the Simplex algorithm encounters difficulty fitting the curve $\beta_o + \beta_1(\Delta h)^p$ through the available data points, it tends to “stiffen” the solution by seeking higher-degree polynomials. This results in regression coefficients $(\beta_o;\beta_1)$ that increase by two and sometimes three orders of magnitude. Likewise, the rate of convergence p increases to values that make no sense, up to 15 and beyond, while the overall MSE remains unchanged. An illustration for the Noh1D problem is provided in Figure 7-4 that shows the results, in terms of MSE and convergence rate values, of several numerical optimizations. Optimization is carried out by limiting the number of iterations to 10, then 20, then 30, etc. The final values of the MSE and solution $(\beta_o;\beta_1;p)$ are recorded when the optimization solver stops at the specified number of iterations. Figure 7-4 illustrates that, beyond 60 iterations, letting the solver iterate does not really improve the MSE while the rate of convergence keeps increasing.

The tendency to over-fit the solution error model can also be observed in Figure 7-5 that pictures the L-shaped curve corresponding to the optimization of the space-only solution error model $e_h = \beta_o + \beta_1(\Delta h)^p$ for the Noh1D problem, density field. Such L-shaped curve is obtained by plotting on the horizontal axis the square of the difference $(p_{Opt}-p_o)^2$ between the optimized rate of convergence and its starting point initialized at $p_o = 1$ and plotting the MSE on the vertical axis. Common practice to avoid over-fitting is to observe the shape of the curve and select the solution at the corner of the “L”, that is, the solution that minimizes the MSE while not stiffening the model too much. Whenever over-fitting is encountered, this strategy is implemented and it results in selecting the solution provided at either 40 or 60 optimization iterations.

Results for the Noh2D problem are summarized next. The Noh problem is solved in two dimensions with computational grids created in a Cartesian (X;Y) coordinate system. As before, the design of computer experiments is a three-level full-factorial array totaling $3^2 = 9$ runs with cell sizes $\Delta h = (2.50.10^{-2} \mid 1.25.10^{-2} \mid 6.25.10^{-3})$ cm and time steps $\Delta t = (5.10^{-3} \mid 10^{-3} \mid 2.10^{-4})$ sec. The spatial discretization results in grids with 80-by-80 = 6,400 cells, 160-by-160 = 25,600 cells, and 320-by-320 = 102,400 cells for the coarse, medium, and fine resolutions, respectively. The computational domain is [0; 2]-by-[0; 2] cm^2. Results are reported at the final simulation time of 0.6 sec. The refinement ratios are $R_H = 2$ in space and $R_T = 5$ in time.

Figure 7-6 illustrates three numerical solutions for the pressure fields obtained with settings $(\Delta h_C;\Delta t_C)$, $(\Delta h_M;\Delta t_M)$, and $(\Delta h_F;\Delta t_F)$. Comparison with the exact solution in Figure 7-6 shows good qualitative agreement. Figures 7-7 and 7-8 picture two numerical solutions for the density fields obtained with the settings $(\Delta h_C;\Delta t_M)$ and $(\Delta h_F;\Delta t_F)$.

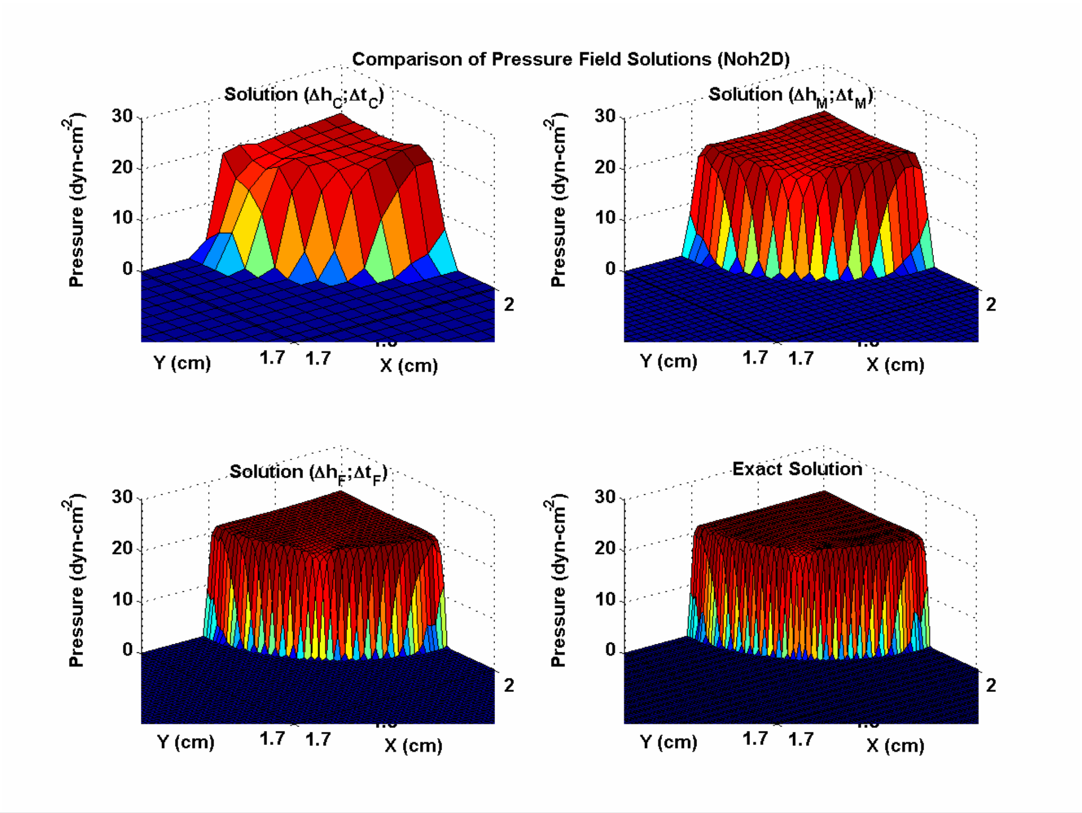


Figure 7-6. Analytical and numerical pressure fields of the Noh2D problem.

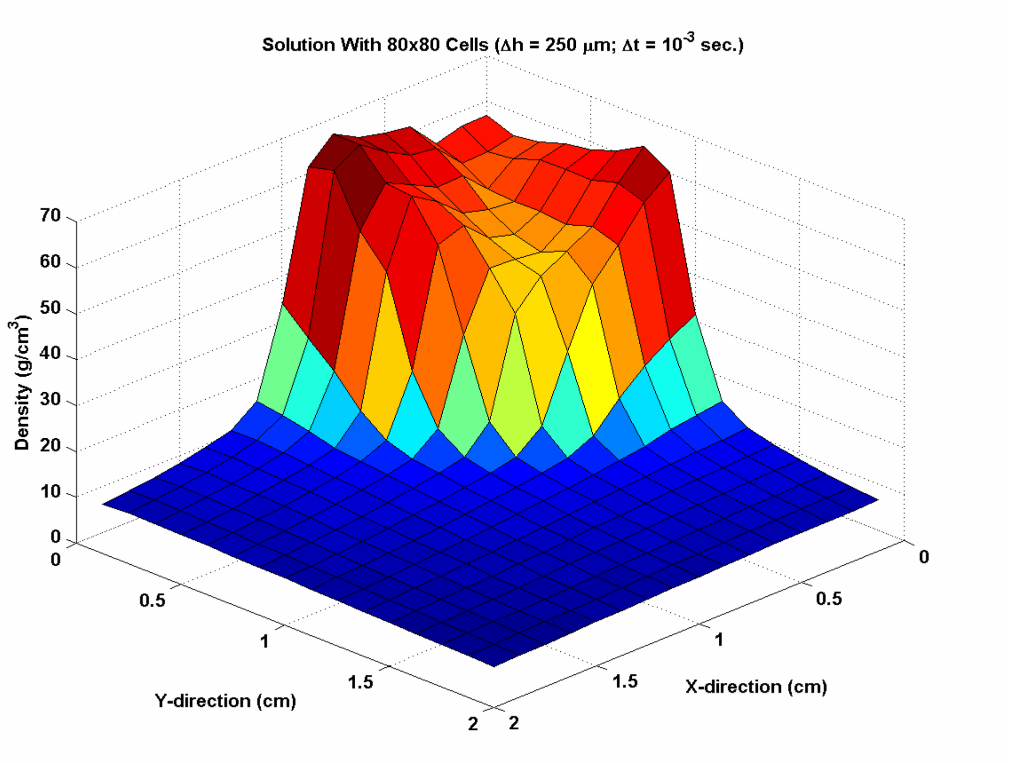


Figure 7-7. Numerical density field of the Noh2D problem obtained with ($\Delta h_C;\Delta t_M$).

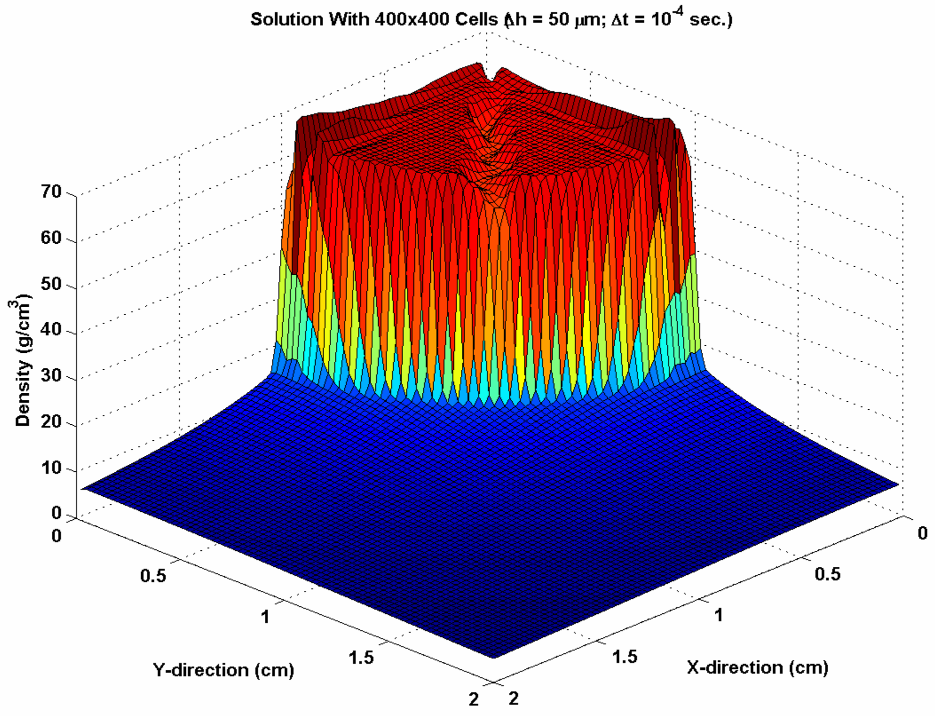


Figure 7-8. Numerical density field of the Noh2D problem obtained with ($\Delta h_F;\Delta t_F$).

Table 7-5 lists the solution convergence error Ansatz models obtained for the pressure fields. As before, the Euclidean L^2 norm is used to calculate the error between numerical and exact solutions over the entire computational domain, that is, $0 \le X \le 2$ cm and $0 \le Y \le 2$ cm.

Table 7-5. Error Ansatz models for the Noh2D problem and pressure field.

Analysis Method	Best Solution Error Model	Mean Square Error (MSE)
Space-only analysis at $\Delta t = 2.10^{-4}$ sec.	$e_h = 2.33 - 10.93(\Delta h)^{1.16}$	0.0%
Space-only analysis at all time steps	$e_h = 1.66 + 0.67(\Delta h)^{0.06}$	2.0%
Space-time analysis at all time steps	$e_h = 1.60 + 0.73(\Delta h)^{0.06} + 1.04(\Delta t)^{1.96} + 1.97(\Delta h)^{1.58}(\Delta t)^{0.55}$	2.0%
Space-time analysis at all time steps, with $\log_{10}$ scaling	$e_h = 1.96 + 0.13(E_H)^{-0.65}$	1.0%

Table 7-5 shows that space-only modeling based on the finest level of time refinement ($\Delta t = \Delta t_F$) captures the first-order convergence of the pressure field in the presence of a discontinuity. Space-time modeling of the solution error seems to indicate that the time main effect and space-time cross interaction are important. Note, however, that the solution error model is constructed with a cross interaction term $(\Delta h)^r(\Delta t)^s$ where the exponents are r = 1.58 and s = 0.55 while the spatial discretization main effect $(\Delta h)^p$ is relatively weak due to an exponent of p = 0.06. It is somewhat suspect to find a strong cross interaction term while the main effect remains relatively weak. This suspicion is confirmed when the transform of variables (7-3) is implemented since the resulting solution error model finds a spatial main effect only.

7.2 Results Obtained With the Sedov Test Problem

The Sedov blast wave problem models the motion of a blast wave in an ideal gas produced by a dense source of energy [7]. The ideal gas has zero temperature and pressure and uniform density. A point-blast energy source is placed at the origin (as a delta function), and the shock produced expands outward as a sphere. Conditions are chosen such that the shock wave is at a radius of R = 1 cm at time T = 1.0 sec. Reference [7] gives details of the problem setting and derivation of the analytical solution.

The Sedov1D problem is solved in perfectly spherical coordinates with the blast wave expending outwards from the origin. The design of computer experiments is a three-level full-factorial array totaling 3^2 = 9 runs with cell sizes $\Delta h = (5.00.10^{-3} \mid 3.3\underline{3}.10^{-3} \mid 2.2\underline{2}.10^{-3})$ cm and time steps $\Delta t = (25.10^{-6} \mid 5.10^{-6} \mid 10^{-6})$ sec. The spatial discretization results in grids with 200, 300, and 450 cells for the coarse, medium, and fine resolutions, respectively. The size of the computational domain is [0; 1] cm. Results are reported at the final simulation time of 1 sec. The refinement ratios are R_H = 1.5 in space and R_T = 5 in time. Results shown are for the density and energy state variables. The solution convergence errors are calculated over the computational domain $0 \leq X \leq 1$ cm with the L^2 norm and scaling ($1/N_{Cells}$) that accounts for the number of cells.

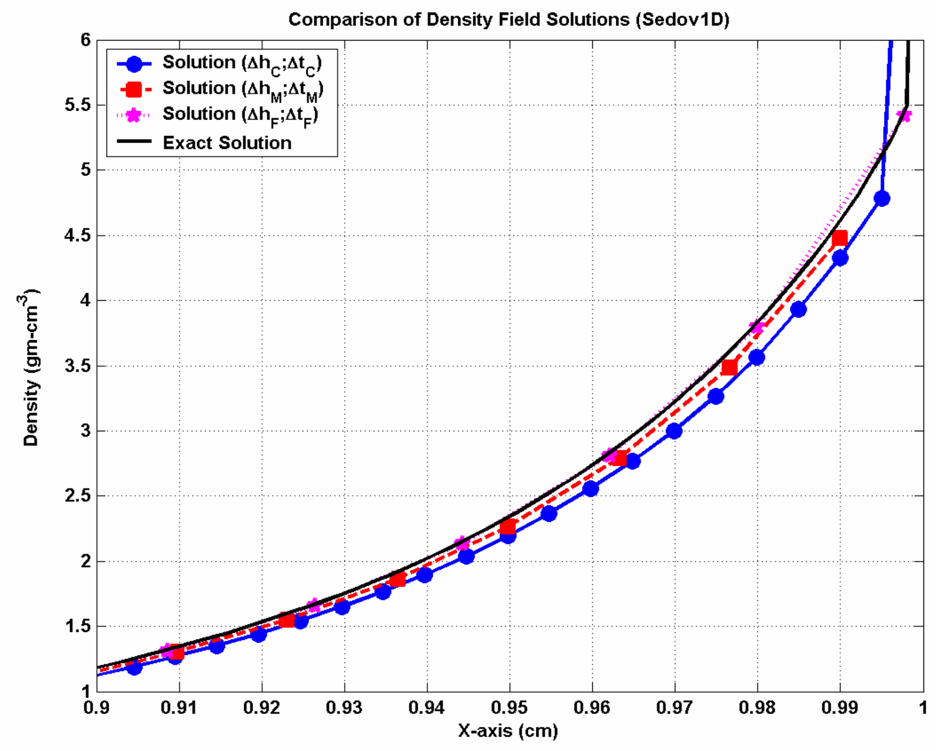


Figure 7-9. Analytical and numerical density fields of the Sedov1D problem.

Figures 7-9 and 7-10 illustrate three numerical solutions for the density and energy fields obtained with settings ($\Delta h_C;\Delta t_C$), ($\Delta h_M;\Delta t_M$), and ($\Delta h_F;\Delta t_F$). Comparison with the exact solutions shows convergence. Agreement is overall good between the solutions provided by the most refined grid and analytical derivation, whether the solution is discontinuous (for the density) or smooth (for the energy).

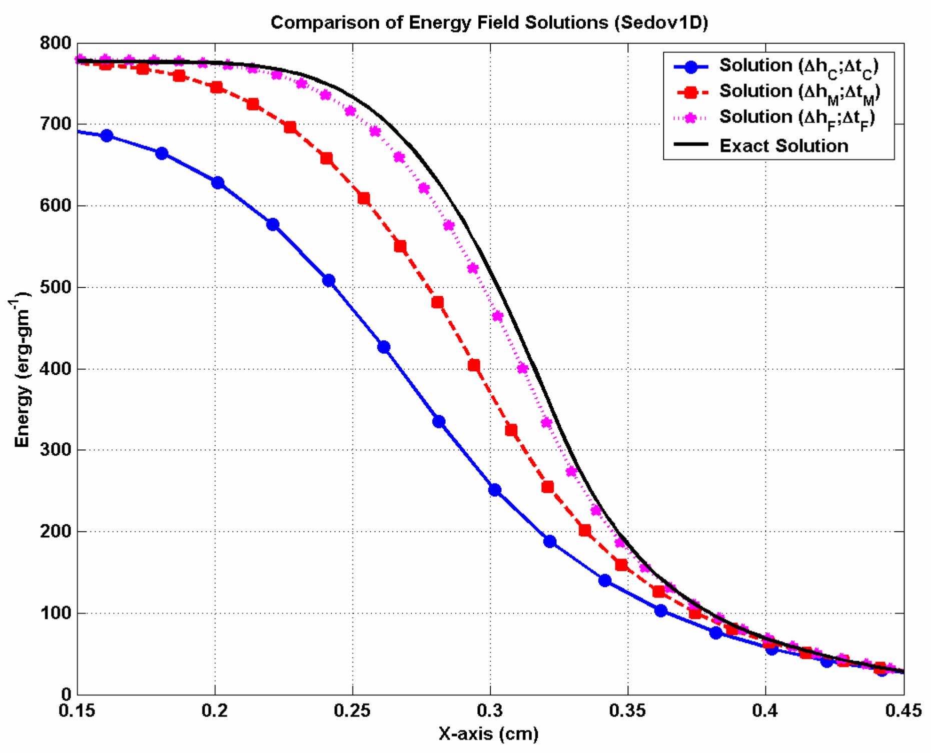


Figure 7-10. Analytical and numerical energy fields of the Sedov1D problem.

Table 7-6 lists the solution convergence error Ansatz models obtained for the density state variable in one dimension (Sedov1D). Table 7-7 lists the density solution error models for the Sedov2D problem, that is, the same problem solved in a two-dimensional domain.

Table 7-6. Error Ansatz models for the Sedov1D problem and density field.

Analysis Method	Best Solution Error Model	Mean Square Error (MSE)
Space-only analysis at $\Delta t = 10^{-6}$ sec.	$e_h = 0.20 - 15.17(\Delta h)^{1.24}$	0.4%
Space-only analysis at all time steps	$e_h = 0.19 + 0.14(\Delta h)^{6.85}$	0.4%
Space-time analysis at all time steps	$e_h = 0.19 + 1.01(\Delta h)^{1.02} + 1.08(\Delta t)^{1.22} + 1.12(\Delta h)^{1.07}(\Delta t)^{1.11}$	0.4%
Space-time analysis at all time steps, with $\log_{10}$ scaling	$e_h = 2.43 + 4.20(E_H)^{-1.56} - 2.09(E_T)^{-6.59} - 6.61(E_H)^{-0.76}(E_T)^{-0.01}$	0.098%

The Sedov2D results are obtained by solving the Euler equations of compressible gas dynamics for the Sedov setting in two dimensions with computational grids created in a Cartesian (X;Y) coordinate system. The design of computer experiments is a three-level full-factorial array totaling $3^2 = 9$ runs with cell sizes $\Delta h = (25.00.10^{-3} \mid 12.50.10^{-3} \mid 6.25.10^{-3})$ cm and time steps $\Delta t = (8.10^{-5} \mid 4.10^{-5} \mid 2.10^{-5})$ sec. The spatial discretization results in grids with 48-by-48 = 2,304 cells, 96-by-96 = 9,216 cells, and 192-by-192 = 36,864 cells for the coarse, medium, and fine resolutions, respectively. The computational domain is [0; 1.2]-by-[0; 1.2] cm^2. Results are reported at the final simulation time of 1 sec. The refinement ratios are $R_H = 2$ in space and $R_T = 2$ in time.

Table 7-7. Error Ansatz models for the Sedov2D problem and density field.

Analysis Method	Best Solution Error Model	Mean Square Error (MSE)
Space-only analysis at $\Delta t = 2.10^{-5}$ sec.	$e_h = 0.90 – 212.67(\Delta h)^{2.22}$	0.0048%
Space-only analysis at all time steps	$e_h = 0.90 – 60.29(\Delta h)^{1.88}$	0.0062%
Space-time analysis at all time steps	$e_h = 0.93 + 5.27(\Delta h)^{2.35} – 6.21(\Delta t)^{1.71} – 0.22(\Delta h)^{0.48}(\Delta t)^{-0.06}$	0.026%
Space-time analysis at all time steps, with $\log_{10}$ scaling	Log_{10} scaling does not improve the results significantly because the above model already fits the data very well.	

Results for the Sedov1D problem in Table 7-6 indicate first-order convergence in space and time, as expected in the presence of a shocked solution for the density field. When simulation results at all time settings (Δt_C, Δt_M, and Δt_F) are used to perform a space-only convergence analysis, a suspicious rate of convergence equal to p = 6.85 is obtained. Rather than being a property of the numerical solver, this large value of the convergence rate is thought be a manifestation of polynomial optimization over-fitting. The variable transform from ($\Delta h;\Delta t$) to ($E_H;E_T$) indicates that the main effect $(\Delta t)^q$ of time discretization is significant to explain the observed convergence solution errors.

Results for the Sedov2D problem in Table 7-7 tend to indicate second-order convergence in space and time. The space-time coupling term $(\Delta h)^r(\Delta t)^s$ is found significant according to its posterior probability even though its contribution to the solution error value is small compared to the contributions of main effects $(\Delta h)^p$ and $(\Delta t)^q$. All solution error models shown in Table 7-7 fit the available data very well, as indicated by MSE values significantly less than 1%. The result of second-order convergence in space and time is to be taken with caution because an erroneous definition of the boundary condition for the Sedov2D calculations is currently being investigated.

7.3 Results Obtained With the Vortex Test Problem

The vortex evolution problem solves in two dimensions the Euler equations of compressible gas dynamics where the mean flow initialized at density ρ_o = 1 gm-cm^{-3}, pressure p_o = 1 dyne-cm^{-2}, and velocity $U_x = U_y$ = 1 cm-sec^{-1} is perturbed by an isentropic vortex [17]. The perturbation is defined in terms of velocity δU_x, δU_y, and temperature δT. It is parameterized by a strength parameter ε whose value is selected to be ε = 5:

$$\delta U_x = \frac{-\varepsilon y}{2\pi} e^{\frac{1}{2}\left(1-R^2\right)}, \quad \delta U_y = \frac{\varepsilon x}{2\pi} e^{\frac{1}{2}\left(1-R^2\right)}$$
$$\delta T = \frac{-(\gamma-1)\varepsilon^2}{8\gamma\pi^2} e^{-\left(1-R^2\right)} \tag{7-5}$$

where R denotes the radius, $R^2 = x^2 + y^2$. Reference [17] gives details of the vortex evolution problem setting and derivation of the analytical solution. The solution is continuous and the perturbation introduced to the mean flow (ε = 5) is strong enough that it excites the non-linearity of Euler equations. The perturbation is not strong enough to develop a shock.

The Vortex2D problem is analyzed with a three-level full-factorial design of computer experiments totaling 3^2 = 9 runs with cell sizes Δh = ($25.00.10^{-2}$ | $8.3\underline{3}.10^{-2}$ | $2.7\underline{7}.10^{-2}$) cm and time steps Δt = (2.10^{-2} | 10^{-2} | 5.10^{-3}) sec. The spatial discretization results in grids with 40-by-40 = 1,600 cells, 120-by-120 = 14,400 cells, and 360-by-360 = 129,600 cells for the coarse, medium, and fine resolutions, respectively. The size of the computational domain is [0; 10]-by-[0; 10] cm^2. The refinement ratios are R_H = 3 in space and R_T = 2 in time. Results shown are for the density state variable at the final simulation time of 100 sec. The solution convergence errors are calculated over the computational domain $0 \le X \le 10$ cm and $0 \le Y \le 10$ cm with the L^2 norm and scaling $(1/N_{Cells})$ that accounts for the number of cells. The exact same settings are used for the wave propagation problem, Wave2D, discussed in Section 7-4.

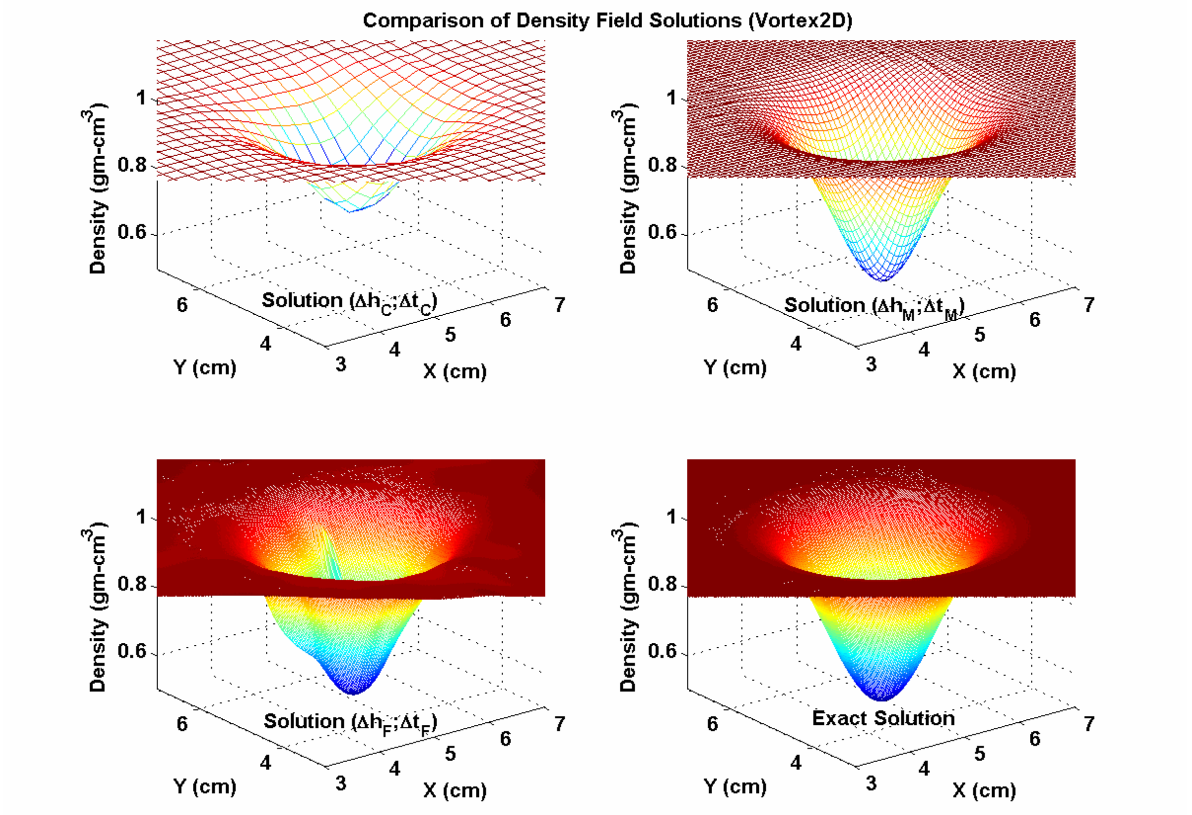


Figure 7-11. Analytical and numerical density fields of the Vortex2D problem.

Figure 7-11 illustrates three numerical solutions for the density state variable obtained with the settings $(\Delta h_C;\Delta t_C)$, $(\Delta h_M;\Delta t_M)$, and $(\Delta h_F;\Delta t_F)$. Comparison with the exact solution seems to indicate convergence as $\Delta h \rightarrow 0$ and $\Delta t \rightarrow 0$.

For this and all other test problems, the constant cell size and time step values are selected such that the CFL condition is satisfied. This involves running the code verification test problem with nominal settings, that is, with the recommended cell size and CFL-limited time steeping, and interrogating the RAGE output dumps to estimate the maximum value of the speed of sound ω_{Sound}. For all test problems analyzed, the maximum value of ω_{Sound} occurs early in the simulation as indicated by time step values Δt that start at 10^{-5} to 10^{-4} sec., then, increase as the number of cycles grows, sometimes by two orders of magnitude. For the Vortex2D and Wave2D problems, the maximum speed of sound is estimated at ω_{Sound} = 1.55 cm-sec^{-1} and the run budget dictates to select settings of cell size and time step pairs $(\Delta h;\Delta t)$ that closely satisfy the CFL condition expressed approximately as:

$$CFL \approx \left(\frac{\Delta t}{\Delta h}\right)\max\left(\omega_{Sound}\right) < 1 \qquad (7\text{-}6)$$

Table 7-8 lists the CFL numbers obtained for each one of the computer runs performed. Run number 3 defined at ($\Delta h_F;\Delta t_C$) violates the CFL condition because it combines the fine cell size to the coarse time resolution. Run 3 resulted in exiting the hydro-code RAGE with an error message. It was decided to perform the analysis based on this data set anyway from which the third run was disregarded.

Table 7-8. CFL numbers estimated for the Vortex2D and Wave2D problems.

Run Number	Cell Size Δh (cm)	Time Step Δt (sec.)	$\omega_{Max} = \Delta h/\Delta t$ Captured by the Grid (cm-sec^{-1})	CFL Number, Equation (7-6)	Run Kept?
1	$25.00.10^{-2}$	2.10^{-2}	12.50	0.12	Yes
2	$8.3\underline{3}.10^{-2}$	2.10^{-2}	$4.1\underline{6}$	0.37	Yes
3	$2.7\underline{7}.10^{-2}$	2.10^{-2}	$1.3\underline{8}$	1.12	No
4	$25.00.10^{-2}$	10^{-2}	25.00	0.06	Yes
5	$8.3\underline{3}.10^{-2}$	10^{-2}	$8.3\underline{3}$	0.18	Yes
6	$2.7\underline{7}.10^{-2}$	10^{-2}	$2.7\underline{7}$	0.56	Yes
7	$25.00.10^{-2}$	5.10^{-3}	50.00	0.03	Yes
8	$8.3\underline{3}.10^{-2}$	5.10^{-3}	$16.6\underline{6}$	0.09	Yes
9	$2.7\underline{7}.10^{-2}$	5.10^{-3}	$5.5\underline{5}$	0.28	Yes

Eliminating the third run has little effect in terms of model fitting because 9 – 1 = 8 computer runs suffice to best-fit space-only models (3 unknowns) or space-time models (7 unknowns when the intercept β_o is eliminated because β_o is equal to the mean solution error, $\beta_o = \bar{e}$). For effect screening, eliminating the third run deteriorates the design of computer experiments in an incomplete full-factorial, which potentially introduces aliasing. Aliasing means that the significance of low-order effects is compounded by the influence of higher-order effects. This is however not believed to be an issue for the Vortex2D and Wave2D problems because, first, the design is "almost" full-factorial (only a single point is missing) and, second, it is verified a posteriori that effect screening for both problems produces no abnormal result.

Table 7-9. Error Ansatz models for the Vortex2D problem and density field.

Analysis Method	Best Solution Error Model	Mean Square Error (MSE)
Space-only analysis at $\Delta t = 5.10^{-3}$ sec.	$e_h = 0.08 + 0.38(\Delta h)^{4.02}$	0.0%
Space-only analysis at all time steps	$e_h = 0.10 - 0.95(\Delta h)^{3.64}$	0.4%
Space-time analysis at all time steps	$e_h = -0.69 + 3.55(\Delta h)^{6.58} + 1.44(\Delta t)^{0.07} - 1.25(\Delta h)^{0.17}(\Delta t)^{0.26}$	0.2%
Space-time analysis at all time steps, with $\log_{10}$ scaling	Log_{10} scaling does not improve the results significantly because the above model already fits the data very well.	

Table 7-9 lists the solution convergence error Ansatz models obtained for the density fields of the Vortex2D problem. Space-only models indicate a rate of convergence close to quadratic, even when all eight simulation runs are fed to the analysis. The main observation from space-time modeling is to confirm that the effect of time step is not significant as indicated by the small values of exponents q = 0.07 and s = 0.26 for the main effect of time discretization $(\Delta t)^q$ and the cross interaction $(\Delta h)^r(\Delta t)^s$, respectively.

7.4 Results Obtained With the Wave Test Problem

The wave propagation problem, denoted as Wave2D, is similar to the previous Vortex2D problem with the exception that no mean flow perturbation is introduced in the initial condition. Because the perturbation parameter is set to $\varepsilon = 0$ in equation (7-5), the Euler equations reduce to a linear wave equation about the density ρ, from which the exact solution can be written analytically. The Euler equations are nevertheless solved for with the hydro-code RAGE. A noticeable difference with the previous vortex evolution problem is that the Wave2D solution is strictly linear since the wave propagation equation is linear and no perturbation of the initial condition is considered. The Wave2D solution is smooth (continuous) and linear at all time steps and over the computational domain.

The same grids as those defined for the Vortex2D problem are used, together with the same constant time steps. Table 7-8 lists the corresponding CFL conditions from which the third computer run is rejected because its grid and constant time step cannot propagate information faster than the maximum observed sound speed of ω_{Sound} = 1.55 cm-sec^{-1}. The analysis mirrors the one performed for the Vortex2D problem with density solutions reported at the simulation time of 100 sec., solution convergence errors calculated over the square computational domain $0 \le X \le 10$ cm and $0 \le Y \le 10$ cm, and L^2 norms scaled to account for the number of cells of each grid.

Figure 7-12 illustrates three numerical solutions for the density state variable obtained with the settings $(\Delta h_C;\Delta t_C)$, $(\Delta h_M;\Delta t_M)$, and $(\Delta h_F;\Delta t_F)$. As before, comparison with the exact solution seems to indicate convergence as $\Delta h \rightarrow 0$ and $\Delta t \rightarrow 0$.

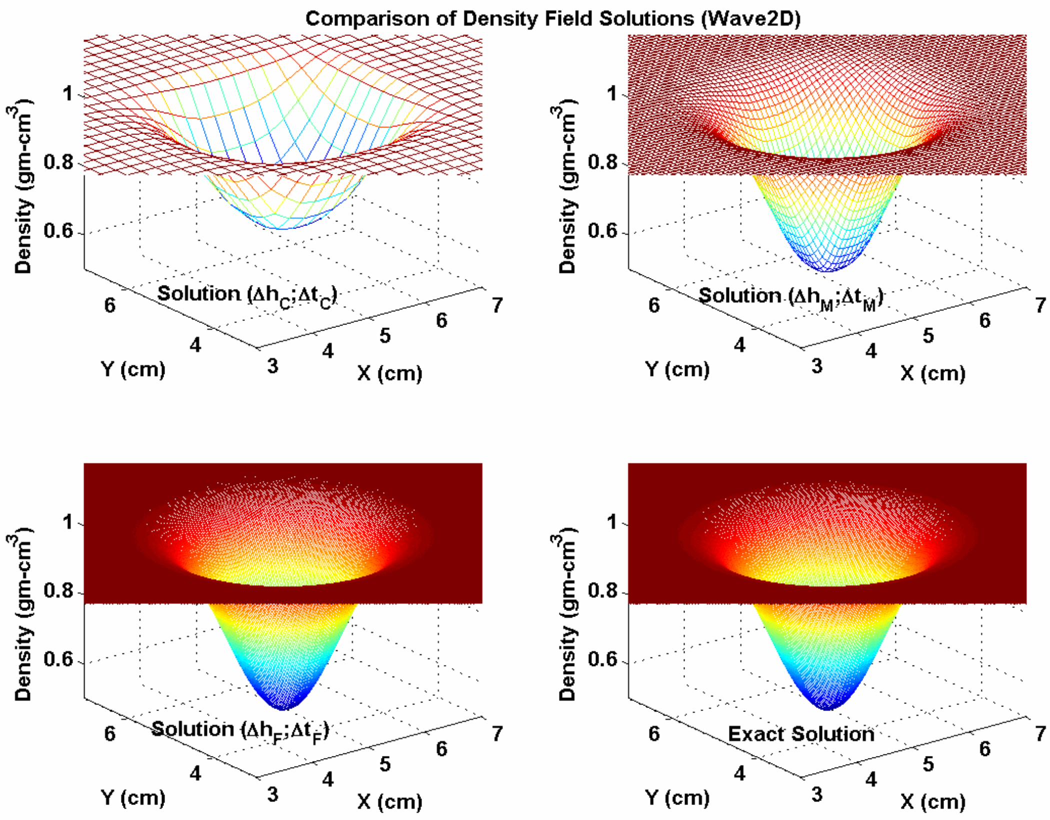


Figure 7-12. Analytical and numerical density fields of the Wave2D problem.

Table 7-10 summarizes the solution convergence error Ansatz models obtained for the density fields of the Wave2D problem. As before, space-only models indicate a quadratic rate of convergence, even when all eight simulation runs are fed to the analysis. Space-time modeling indicates quadratic convergence in space, quadratic (almost cubic) convergence in time, and the presence of a space-time coupling term although this cross interaction effect is influential to a lesser extent.

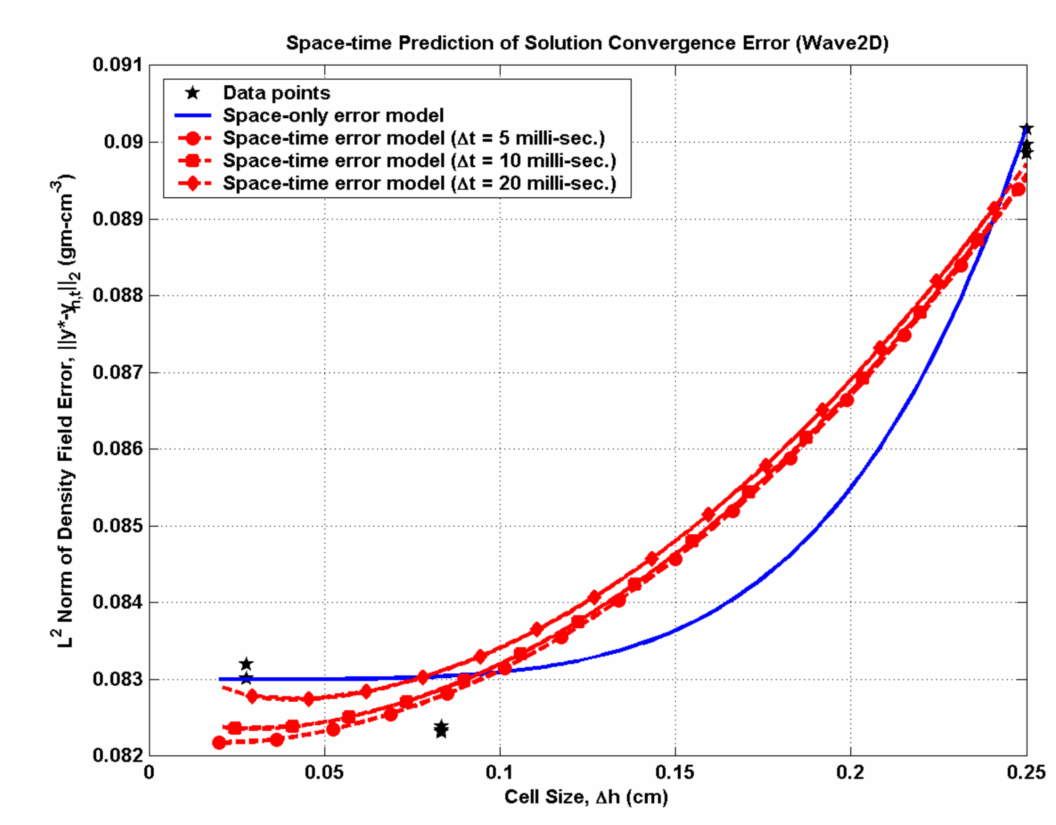


Figure 7-13. Forecasting of solution error with space-only and space-time models.

Table 7-10. Error Ansatz models for the Wave2D problem and density field.

Analysis Method	Best Solution Error Model	Mean Square Error (MSE)
Space-only analysis at $\Delta t = 5.10^{-3}$ sec.	$e_h = 0.08 + 5.12(\Delta h)^{4.74}$	0.0%
Space-only analysis at all time steps	$e_h = 0.08 + 4.49(\Delta h)^{4.63}$	0.0%
Space-time analysis at all time steps	$e_h = 0.08 + 0.14(\Delta h)^{2.13} + 3.43(\Delta t)^{2.97} + 0.01(\Delta h)^{-0.63}(\Delta t)^{1.30}$	0.0%
Space-time analysis at all time steps, with $\log_{10}$ scaling	Log_{10} scaling does not improve the results significantly because the above model already fits the data very well.	

Figure 7-13 compares graphically the solution error values predicted by the space-only and space-time models of Table 7-10. The first space-only model, $e_h = 0.08 + 5.12(\Delta h)^{4.74}$, is used to generate the data shown. Because the space-only model cannot, by definition, account for Δt, it is represented as a single curve in Figure 7-13. The space-time model, on the other hand, is evaluated at $\Delta t = 5.10^{-3}$ sec., $\Delta t = 10.10^{-3}$ sec., and $\Delta t = 20.10^{-3}$ sec., which produces three slightly different curves. The figure illustrates that predictions of solution convergence errors at settings (Δh;Δt) that have not been used to best-fit the error model can differ depending on the functional form of the model and its inclusion or not of time discretization.

One final observation is that the space-time coupling effects $(\Delta h)^r(\Delta t)^s$ for this and other test problems (Noh1D, Vortex2D, Wave2D) exhibit exponents of opposite signs ($r > 0$ while $s < 0$ or the opposite combination $r < 0$ while $s > 0$). This may be an indication that the cell size-to-time step ratio $(\Delta h/\Delta t)$ is an important explanatory variable. Although this was not done here, the analysis could be repeated to include this third variable in addition to cell size and time step in an attempt to improve the quality of fit to the data. Physically, the explanatory variable $(\Delta h/\Delta t)$ makes sense since it represents the maximum speed at which a cell can propagate information, at least locally, to the neighboring cells, which may be a variable of importance in the presence of a discontinuity or strong gradient.

7.5 Short Description of the FEAST Toolbox

The results presented in this report have been obtained using a dedicated software package called FEAST (Fitting Error Ansatz in Space and Time). FEAST is a collection of MATLAB™ functions currently specialized to the four code verification test problems. For a given test problem, FEAST provides a design of computer experiments (full factorial, two-level orthogonal array, Monte Carlo, or Latin Hypercube sampling); writes the corresponding input decks; and writes a macro-command to execute the individual runs on the QSC machine. Once the runs have been completed and transferred back from QSC, FEAST uploads the text dump files (obtained after post-processing the ASCII dump files of RAGE with AMHCTools [8]); extracts the density, energy, pressure, velocity, and temperature solution fields; computes the error norms; and performs first-order ANOVA, linear interaction ANOVA, deterministic model fitting, or statistical model fitting.

About 5,000+ lines of codes have been written that can easily be extended to include other verification test problems, design of computer experiments, or analysis techniques. Figure 7-14 illustrates some of the FEAST code written to drive the analysis. One of the future tasks will be to extend the current capabilities of the FEAST toolkit, for example, by integrating solvers to estimate point-wise solution convergence error Ansatz models [24], or, conversely, to work with computer scientists to transfer the capabilities of FEAST to a third-party ASC (or other) testing harness software.

```
%...STEP 1: Define the code verification problem to analyze
[Definition] = feast_def(1,[],{Problem;TypeDOE;nSamples});

%...STEP 2: Define the design of computer experiments
[Output] = feast_doe(Definition,Options);
%...Update the problem definition structure
[Definition] = feast_def(2,Definition,Output,Options);

%...STEP 3: Create the input decks
[Output] = feast_input(Definition,Options);
%...Update the problem definition structure
[Definition] = feast_def(3,Definition,Output,Options);

%...STEP 4: Create the execution commands
[Output] = feast_run(Definition,Options);
%...Update the problem definition structure
[Definition] = feast_def(4,Definition,Output,Options);

%...STEP 5: Perform the analysis
%*/*/* Transfer files and execute on QSC */*/*

%...STEP 6: Upload results back in MATLAB memory
[Output] = feast_upload(Definition,Options);
%...Update the problem definition structure
[Definition] = feast_def(6,Definition,Output,Options);

%...STEP 7: Fit error Ansatz models
[Output] = feast_fit(Definition,Options);
%...Update the problem definition structure
[Definition] = feast_def(7,Definition,Output,Options);
```

Figure 7-14. Sample of FEAST instructions for driving an error Ansatz analysis.

This page is left blank intentionally.

8. Conclusion

Summary: The overall conclusion of the study is that including time discretization effects and space-time coupling terms may be critical to predict solution convergence error and quantify numerical uncertainty. Because this clearly depends on the code and physics being simulated, developing a general-purpose framework for the formulation of solution convergence error Ansatz models is necessary. One such approach is proposed that relies on the design of computer experiments, analysis-of-variance, and statistical effect screening and model fitting. Directions for future research and development are identified.

This report discusses work performed in support of the fiscal year 2005 Code Verification project of the Advanced Scientific Computing (ASC) program at Los Alamos National Laboratory to investigate the correctness of error Ansatz models for solution verification. It is proposed to develop error Ansatz models that include, if appropriate, time discretization effects and terms that accommodate space-time coupling effects.

The overall conclusion, based on the code verification test problems investigated, is that the coupling between cell size and time step greatly matters to forecast numerical solution uncertainty. Another contribution of the study is to suggest a general-purpose procedure based on the design of computer experiments, analysis-of-variance, and statistical model fitting to formulate non-linear error Ansatz models for solution convergence.

Advantages of this approach are that it can be: applied in a somewhat black-box mode, that is, independently of the code or verification test problem; automated to a great extent; and extended to include non-constant cell sizes (to handle adaptive mesh refinement), non-constant time steps (to handle stability-limited runs), and other parameters that control the discretization of partial differential equations.

Limitations of the study are that it is based on: specific verification test problems for which the exact solutions of the continuous equations are known (the more difficult case of solution self-convergence is not addressed); a single Eulerian code developed under the Los Alamos Code Project "Crestone" (other codes are not considered); uniform grids (adaptively refined grids are not considered); and constant time stepping (simulations are not analyzed with stability-limited time steps like it is customary in computational physics).

Directions identified for future research and development are listed below in no particular order:

1) Extend the investigation to non-uniform meshes and non-uniform time stepping, such as those encountered in adaptive mesh refinement calculations with stability-limited time stepping. One issue to address is the characterization of the cell size and time step when they vary, potentially, from cell-to-cell and cycle-to-cycle. A likely roadblock, to be discussed with the code development team, is the ability to track the evolution and access the values of time steps and cell sizes for the entire grid and at each cycle of the simulation.

2) Extend the investigation to other physics codes, whether Eulerian or Lagrangian, developed under the Los Alamos Code Projects "Crestone" and "Shavano". Another interesting investigation would be to perform back-to-back comparisons of error Ansatz models for solution convergence between ASC and "legacy" physics codes.

3) Incorporate the FEAST software developed for Fitting Error Ansatz Models in Space and Time into an automated analysis and testing harness, for example, the Pinocchio project (G. Hrbek, Principal Investigator) or the AMHCTools software (J. Grove, Principal Investigator).

4) Integrate the formulation of error Ansatz models to the case of point-wise estimation of solution convergence for code verification and/or solution self-convergence. In the case of solution self-convergence, one issue is to extend the space-only Richardson extrapolation to the case of space-time equations.

5) Improve the statistical capabilities of the FEAST toolbox. They are currently limited to a few designs of computer experiments and main effect analysis-of-variance. The polynomial fitting functions are limited to quadratic polynomials up to four variables. These capabilities suffice to deal with uniform grids and forced (constant) time stepping. Handling adaptively refined grids and stability-limited time stepping will require extension of the polynomial functions to more than four variables to be able, for example, to study the effects on solution error of a minimum cell size, maximum cell size, minimum time step, maximum time step, and other variables of the discretization or control parameters of the numerical simulation.

6) Extend the approach proposed to the challenging case of solution self-convergence where the exact solution of the continuous equations is unknown. The issue is that numerical solutions obtained from multiple grids are usually extrapolated to calculate a “reference” that estimates the unknown solution. Any extrapolation implicitly assumes the functional form of the error Ansatz equation (such as linear and function of cell size only), which strongly couples the calculation of a reference solution to the formulation of the error Ansatz model.

7) Propagate uncertainty from other control parameters of the simulation, for example, discrete flags that define the numerical algorithms or continuous parameters that control the numerical dissipation, accuracy, stability, etc. This would take a step towards the quantification of total numerical uncertainty, whether it originates from lack of solution convergence or lack-of-knowledge about control parameters of the code. The statistical analysis techniques proposed in the current study can provide the practical means by which numerical uncertainty is quantified for Verification and Validation (V&V).

8) Investigate the diagnosis of solution self-convergence using higher-order statistical moments of the lack-of-fit between predictions of an error Ansatz equation and the observed errors. It might be possible, not only to develop novel approaches to assess self-convergence (of lack thereof), but also to derive statistically rigorous uncertainty bounds for the unknown solution of the continuous equations.

9) Demonstrate the extent to which error Ansatz models can be used in a predictive mode. Having developed a single one or a family of error Ansatz models, apply it/them in a predictive mode to forecast the level of numerical solution uncertainty given simulation parameters such as cell size, time step, etc. Conversely, estimate the requirements in terms of mesh density, number of cycles, etc., such that the solution error is guaranteed not to exceed a given level.

Acknowledgements

This work was performed by the Los Alamos National Laboratory (LANL), which is operated by the University of California for the National Nuclear Security Administration of the United States Department of Energy under contract W-7405-ENG-36. The author acknowledges the support of Scott Doebling, PADNWP, manager of the ASC Verification and Validation (V&V) program at LANL. The author also wishes to express his gratitude to members of the ASC V&V Code Verification project at LANL: Jerry S. Brock, X-7; John W. Grove, CCS-2; Shengtai Li, T-7; James R. Kamm, X-7; Bill Rider, X-7; Mikhail J. Shashkov, T-7; and David Palmer Smitherman, X-7. Their encouragements and scientific interest in this work are greatly appreciated.

This page is left blank intentionally.

References

[1] Ainsworth, M., Oden, J.T., **A Posterior Error Estimation in Finite Element Analysis**, Wiley Inter-science Series in Pure and Applied Mathematics, John Wiley & Sons, Inc., New York City, New York, 2000, pp. 16-23.

[2] Archer, B., Betlach, T., Gittings, M., Scovel, C., Simmons, K., Steinkamp, M., Weeks, D., Weaver, R., **RAGE Users Manual**, *Technical Report*, Los Alamos National Laboratory, Los Alamos, New Mexico, October 2004. LA-CP-04-0423.

[3] Axford, R.A., "Solutions of the Noh Problem for Various Equations of State Using Lie Groups, *Technical Report*, Los Alamos National Laboratory, Los Alamos, New Mexico, August 1998. LA-13497.

[4] Booker, J.M., "A Whirlwind Tour of Statistical Methods in Structural Dynamics," *23rd SEM International Modal Analysis Conference*, Orlando, Florida, February 2005.

[5] Brock, J.S., "ASC Level-2 Milestone Plan: Code Verification, Calculation Verification, Solution-error Analysis, and Test-problem Development for LANL Physics-simulation Codes," *Technical Report of the ASC Program and Code Verification Project*, Los Alamos National Laboratory, Los Alamos, New Mexico, May 2005. LA-UR-05-4212.

[6] Buechler, M., McCarty, A., Reding, D., Maupin, R.D., "Explicit Finite Element Code Verification Problems," *22nd SEM International Modal Analysis Conference*, Dearborn, Michigan, January 2004. LA-UR-03-4603.

[7] Gisler, G., "Two-dimensional Convergence Study of the Noh and Sedov Problems With RAGE: Uniform an Adaptive Grids," *Technical Report of the ASC Program and Code Verification Project*, Los Alamos National Laboratory, Los Alamos, New Mexico, October 2005. LA-UR-05-XXXX.

[8] Grove, J.W., "AMHCTools User Guide and Programming Manual," *Technical Report of the ASC Program and Code Verification Project*, Los Alamos National Laboratory, Los Alamos, New Mexico, October 2005, LA-UR-05-7425. Available as Los Alamos National Laboratory Code Documentation LA-CC-05-052.

[9] Hasselman, T.K., Anderson, M.C., Wenshui, G., "Principal Components Analysis for Nonlinear Model Correlation, Updating and Uncertainty Evaluation," *16th SEM International Modal Analysis Conference*, February 1998, pp. 664-651.

[10] Hemez, F.M., Private communication with Jane Booker, *Record Storage Book 1*, Los Alamos National Laboratory, Los Alamos, New Mexico, January 2005, pp. 77-78.

[11] Hemez, F.M., Brock, J.S., Kamm, J.R., "Non-linear Error Ansatz Models in Space and Time for Solution Verification," *1st Non-deterministic Approaches (NDA) Conference and 47th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials (SDM) Conference*, Newport, Rhode Island, May 2006. LA-UR-05-2664.

[12] Hutchens, G.J., Brock, J.S., Hrbek, G.H., "Automated Convergence Analysis of an ASC Shavano Project Code Using Noh's Hydro-dynamic Test Problem," *Technical Report of the ASC Program and Code Verification Project*, Los Alamos National Laboratory, Los Alamos, New Mexico, September 2005. LA-UR-05-7408.

[13] Kamm, J.R., Rider, W.J., Brock, J.S., "Consistent Metrics for Code Verification," *Technical Report*, Los Alamos National Laboratory, Los Alamos, New Mexico, June 2002. LA-UR-02-3794.

[14] Kamm, J.R., Rider, W.J., Brock, J.S., "Combined Space and Time Convergence Analyses of a Compressible Flow Algorithm," *16th AIAA Computational Fluid Dynamics Conference*, Orlando, Florida, July 2003. LA-UR-03-2628.

[15] Kerschen, G., Golinval, J.C., Hemez, F.M., "Bayesian Model Screening for the Identification of Non-linear Mechanical Structures," *ASME Journal of Vibration and Acoustics*, Vol. 125, July 2003, pp. 389-397. LA-UR-02-2787.

[16] Knoll, D.A., Chacon, L., Margolin, L.G., Mousseau, V.A., "On Balanced Approximations for Time Integration of Multiple Time Scale Systems," *Journal of Computational Physics*, Vol. 185, No. 2, 2003, pp. 583-611.

[17] Li, S., Rider, W.J., Shashkov, M.J., "Two-dimensional Convergence Study for Problems With Exact Solution: Uniform and Adaptive Grids," *Technical Report of the ASC Program and Code Verification Project*, Los Alamos National Laboratory, Los Alamos, New Mexico, October 2005. LA-UR-05-7985.

[18] Mottershead, J.E., Friswell, M.I., "Model Updating in Structural Dynamics: A Survey," *Journal of Sound and Vibration*, Vol. 162, No. 2, 1993, pp. 347-375.

[19] Noh, W.F., "Errors for Calculations of Strong Shocks Using an Artificial Viscosity and an Artificial Heat-flux," *Journal of Computational Physics*, Vol. 72, No. 1, 1987, pp. 78-120.

[20] Rider, W.J., "Revisiting Wall Heating," *Journal of Computational Physics*, Vol. 162, No. 2, August 2000, pp. 395-410.

[21] Roache, P.J., "Perspective: A Method for Uniform Reporting of Grid Refinement Studies," *ASME Journal of Fluids Engineering*, Vol. 116, September 1994, pp. 405-413.

[22] Roache, P.J., **Verification in Computational Science and Engineering**, Hermosa Publishers, Albuquerque, New Mexico, 1998.

[23] Salari, K., Knupp, P., "Code Verification by the Method of Manufactured Solutions," *Technical Report*, Sandia National Laboratories, Albuquerque, New Mexico, 2000. SAND-2000-1444.

[24] Smitherman, D.P., Kamm, J.R., Brock, J.S., "Calculation Verification: Pointwise Estimation of Solutions and Their Method-associated Numerical Error," *Technical Report of the ASC Program and Code Verification Project*, Los Alamos National Laboratory, Los Alamos, New Mexico, October 2005, LA-UR-05-8002.

[25] Trucano, T., *Personal Communication*, February 2005.

[26] Walter, E., Pronzato, L., **Identification of Parametric Models From Experimental Data**, Springer-Verlag, Berlin, Germany, 1997.